平成 24 年度　修士論文

LTE ハンドオーバーを想定した HTTP Streaming 性能評価実験

早稲田大学大学院　基幹理工学研究科　情報理工学専攻

5111B085-2

野﨑　寛也

指導　甲藤二郎 教授

2013 年 2 月 8 日

指導教授印	受付印

目次

第１章　序論

1.1　はじめに

現在、インターネットをはじめとするIP網の大規模化が進み、ネットワークは年々拡大・発展し続けてきている。例えば、アクセス系のネットワークでは、FTTH(Fiber to The Home)の利用が一般的となり、無線通信に関しては、MIMO等の技術を利用し最大600[Mbit/s]の伝送速度を実現する802.11nが一般家庭でも使われるなど高速化している。また、クラウドコンピューティングと呼ばれるネットワークを経由してコンピュータ処理の実行や、アプリケーションを利用する技術も年々発展しており、ネットワークはこれからもさらに多くのトラフィックが流れ、大規模化していくと考えられる。

高速化・大容量化の一例としてLTE(Long Term Evolution)が注目を集めている。LTEは従来の3Gと比べ、通信速度・周波数の利用効率を向上させる事ができ、周波数帯域幅の柔軟性が高く、また、接続遅延等の遅延時間も短縮する事ができるとされている。そのため、スマートフォンの利用が一般的になった事やトラフィック量が増加していく状況もあり、普及が進んでいる。加えて、2013年1月から3月の間に下り通信速度が最大112.5[Mbit/s]の新サービスが検討されるなど、今後も急速に発展すると考えられる。

一方、近年、インターネットでは、YouTube[1]やニコニコ動画[2]などの様々なアプリケーションを用いて、動画像サービスの提供が行われている。このような動画像データ(ストリーミングデータ)は、従来、RTP/UDPを用いて配信するのが主流であった。しかし、インターネットの広域化とCDN (Content Delivery Network) の普及や、HTTPを用いれば、キャッシュを利用できる事やファイアフォールやNATを簡単に越えられること、サーバーを簡単に用意できるなどの利点がある事から、HTTP/TCPを用いての配信が主流になりつつある。

さらに、Apple社の開発したiPhoneやiPadの普及と共に、Apple社が提案したHTTP Live Streaming[3]やAdobe社の提案したHTTP Dynamic Streaming[4]が注目を集めている。これらの技術は、ネットワーク環境の変化に関わらず、あらかじめ決められたビットレートの動画を常にユーザーに提供しようとする。そのため、急に遅延が大きくなる、もしくはパケット損失率が高くなるといったネットワーク環境の悪化に対応する事ができずに、動画の再生が途切れ途切れになる場合や、動画が長時間停止してしまう事が考えられる[5]。また、一般的にユーザーは動画の再生が停止する、途切れるといった状況を好ましく思わない事がわかっている[6]。

そこで、近年注目を集めているのが、DASH(Dynamic Adaptive Streaming over HTTP)[7]と呼ばれる技術である。この技術はネットワーク環境の変化に応じて、配信するストリーミングデータのビットレート等を動的に変化させるため、動画の再生が停止する、途切れるといった状況を防ぐことができる。また、動画の音声の言語を変化させる事や、字幕の有無なども設定する事ができ、QoEやQoSの向上も図る事ができる。

これらの点をふまえ、本研究では、LTEハンドオーバーが発生する状況下において、HTTP Live StreamingやDASHを行った場合、どのような問題点や課題が発生するかシミュレーション実験を行い、調査している。その結果、DASHを利用した場合の方が、HTTP Live Streamingを利用する場合に比べて、スループット、動画のダウンロードにかかる時間といった観点から優れているという事がわかった。

1.2　本論文の構成

第2章では、TCPの輻輳制御方式について述べる。

第3章では、ストリーミング技術について述べる。

第4章では、LTEについて述べる。

第5章では、シミュレーション実験について述べる。

第6章では、まとめと今後の展望について述べる。

第 2 章　TCP の輻輳制御方式

2.1　TCP による輻輳制御

TCP の輻輳制御は基本的にスロースタートフェーズと輻輳回避フェーズの 2 フェーズから構成されている。TCP の種類によって、輻輳回避フェーズでの輻輳制御方法は異なり、その違いがそれぞれの TCP を特徴づけていると言える。ここでは 1988 年に発表され、TCP の輻輳制御方式の基本となっている TCP-Tahoe[8]のスロースタートフェーズ、輻輳回避フェーズについて述べる。

2.1.1　スロースタートフェーズ

スロースタートフェーズは、新たに通信を開始する場合、もしくは、タイムアウトやパケット損失等で輻輳ウィンドウサイズが、スロースタート閾値を下回っている場合に利用する。スロースタート開始時の輻輳ウィンドウサイズは 1 であり、確認応答(ACK)を 1 つ受信するたびに 1MSS(Maximum Segment Size)ずつ輻輳ウィンドウサイズを増加させる。増加前の輻輳ウィンドウサイズを cwnd、増加後の輻輳ウィンドウサイズを$\mathrm{cwnd}_{\mathrm{new}}$、最大セグメント長を 1MSS とすると、以下のような式で表すことができる

$$\mathrm{cwnd}_{\mathrm{new}} = \mathrm{cwnd} + 1\mathrm{MSS} \tag{2.1}$$

図 2.1 からわかるようにスロースタート開始時は輻輳ウィンドウサイズが 1 であるので、データセグメントを 1 つ送信する。その際、ACK を 1 つ受信するので、次の RTT2 での輻輳ウィンドウサイズは 2 に増加する。RTT2 ではデータセグメントを 2 つ送信することができるので、2 つの ACK を受信する。すると、次の RTT3 での輻輳ウィンドウサイズは 4 に増加する。同様にして次の RTT3 では 4 つの ACK を受信して、輻輳ウィンドウサイズは 8 に増加する。このようにして、スロースタートフェーズでは輻輳ウィンドウサイズが指数的に増加していく。

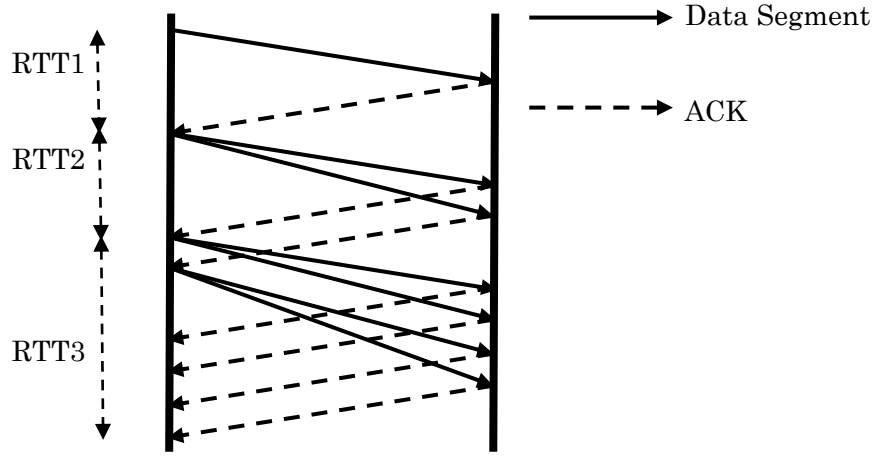


図 2.1 スロースタートフェーズ

2.1.2 輻輳回避フェーズ

輻輳回避フェーズでは、1 つの ACK を受信するたびに、輻輳ウィンドウサイズを 1/輻輳ウィンドウサイズ分だけ、増加させる。1RTT に輻輳ウィンドウサイズ分の確認応答を受け取るので、1RTT に 1MSS 分増加する事になり、スロースタートと異なり、線形的な増加になる。増加前の輻輳ウィンドウサイズを cwnd、増加後の輻輳ウィンドウサイズを cwnd_new、最大セグメント長を 1MSS とすると、ACK を 1 つ受信したときの輻輳ウィンドウの増加は以下のような式で表すことができる。

$$\mathrm{cwnd}_{\mathrm{new}} = \mathrm{cwnd} + \frac{1\mathrm{MSS}}{\mathrm{cwnd}} \tag{2.2}$$

図 2.2 のように、RTT1 でデータセグメントが 2 つ送信されたとすると、ACK を 2 つ受信することになる。1 つの ACK ごとに 1/2 ずつ輻輳ウィンドウサイズが増加するので、次の RTT3 での輻輳ウィンドウサイズは 2+1 で 3 になる。同様にして RTT2 では 3 つの ACK を受信するので、RTT3 での輻輳ウィンドウサイズは 4 に増加する。このようにして、輻輳回避フェーズでは、線形的に輻輳ウィンドウサイズが増加していく。

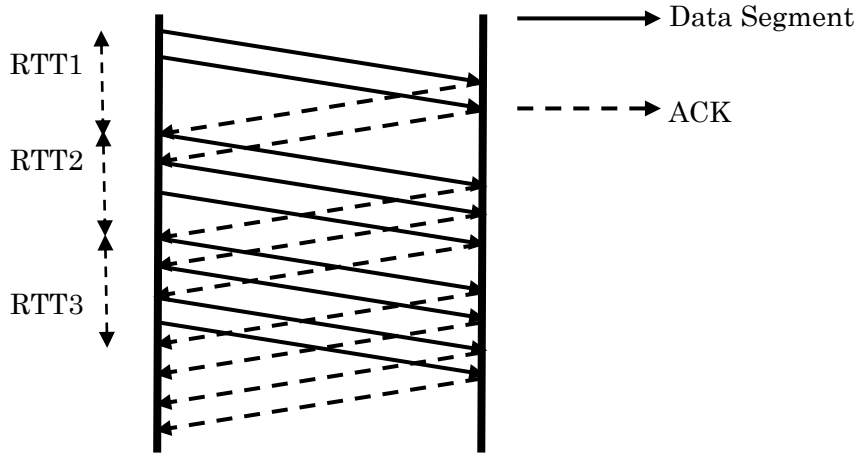


図 2.2 輻輳回避フェーズ

2.1.3 通信フェーズの移行

スロースタートフェーズから輻輳回避フェーズへの移行は、スロースタート閾値(ssthresh)を境にして行われる。輻輳ウィンドウサイズが ssthresh より小さい場合はスロースタートフェーズであり、輻輳ウィンドウサイズが ssthresh を超えた場合に輻輳回避フェーズに移行する。TCP はパケット損失やタイムアウトの場合に、輻輳ウィンドウサイズを一時的に減少させ、輻輳崩壊を避けると同時に、スロースタート閾値をパケット損失時の輻輳ウィンドウサイズの半分に設定する。図 2.3 に TCP-Tahoe の輻輳ウィンドウサイズの変化の例を示す。

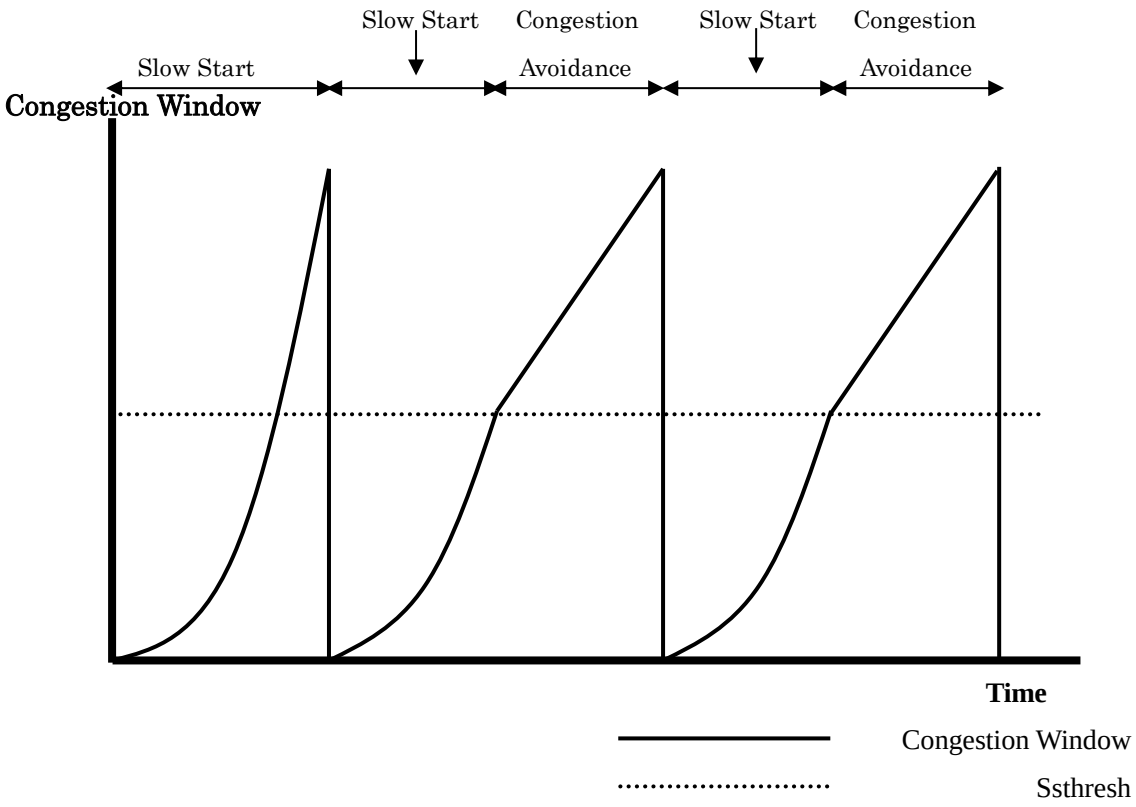


図 2.3 TCP-Tahoe の輻輳ウィンドウ変化

図 2.3 から、スロースタートフェーズで通信が開始され、パケット損失が起こるまで、輻輳ウィンドウサイズを指数的の増加させていることがわかる。やがて、輻輳ウィンドウサイズがネットワークの許容量を超えてしまうと、パケット損失が発生し、輻輳ウィンドウサイズを 1 まで、減少し、スロースタート閾値を輻輳ウィンドウサイズの半分に設定している。パケット損失後は、スロースタート閾値まで、スロースタートフェーズで通信を行い、スロースタート閾値を超えてからは、輻輳回避フェーズで通信を行っている。このようにして、TCP-Tahoe では輻輳ウィンドウサイズが変化する。

2.2 Standard TCP

今日までに標準的に用いられてきた方式で、多くの OS に搭載されている。基本である TCP-Tahoe の欠点を改善したものが多い。

2.2.1 TCP-Reno[9]

TCP-Tahoe では、パケット損失などが発生した場合、輻輳ウィンドウサイズを 1 まで減少させ、またスロースタートからデータ転送をやり直す。この手法では、輻輳ウィンドウサイズが、パケット損失が発生した時の大きさまで回復するのに時間がかかり、結果、転

送性能が低くなるという欠点がある。TCP-Reno ではこの欠点を克服すべく、TCP-Tahoe の輻輳制御に、高速リカバリアルゴリズムを導入している。高速リカバリアルゴリズムは、パケット損失時、輻輳ウィンドウサイズを 1 まで下げずに半分の大きさにする。そして、スロースタートフェーズからやり直さずに、パケット損失後は、輻輳回避フェーズに移る。なお、重複確認応答を 3 回受信した場合をパケット損失と判断する。また、再送タイムアウトの場合は、TCP-Tahoe と同様に、輻輳ウィンドウサイズを 1 まで減少させる。

図 2.4 は TCP-Reno の輻輳ウィンドウサイズの変化を示したものである。

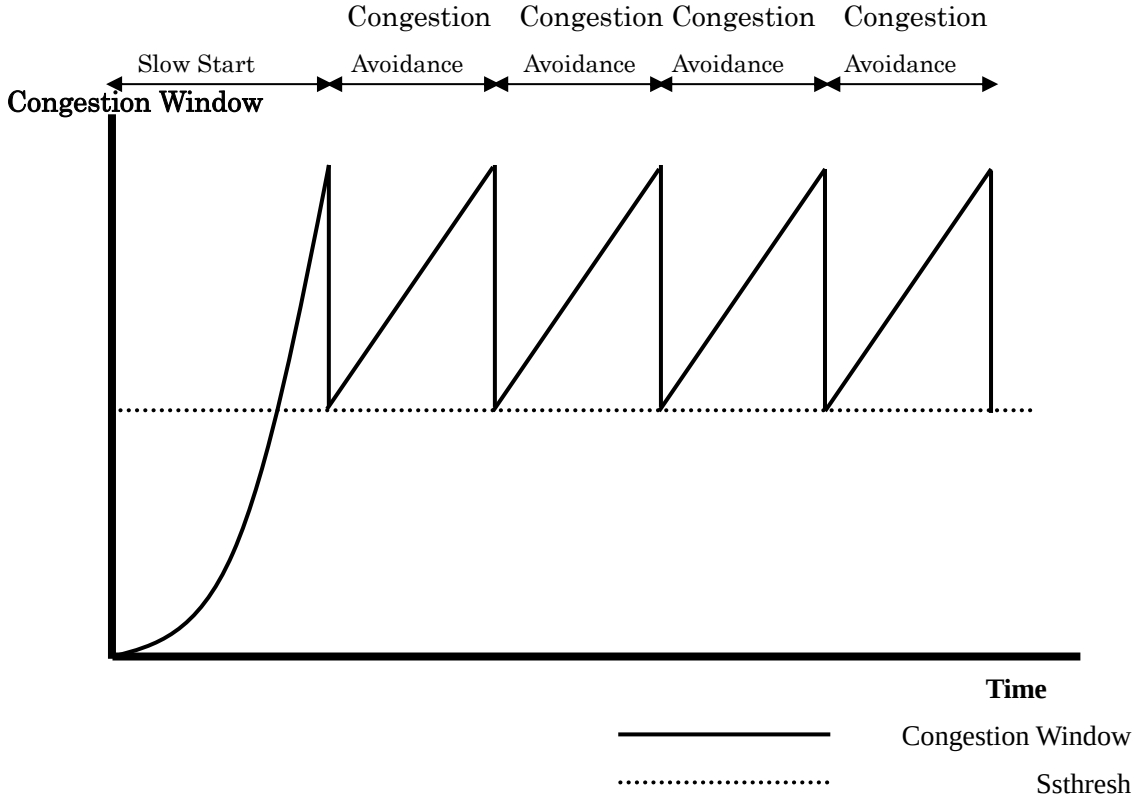


図 2.4 TCP-Reno の輻輳ウィンドウ変化

図 2.4 から、パケット損失してもスロースタートフェーズでなく、輻輳回避フェーズに移行している事がわかる。そのため、パケット損失時の転送効率は TCP-Tahoe に比べて良いと言える。しかし、輻輳ウィンドウサイズにおける変動が激しく、帯域を有効に活用できないとされる。

2.2.2 TCP-NewReno[10]

TCP-Reno と輻輳制御はあまり変わりがないが、重複確認応答を 3 回受信した後の再送制御に改良を加えている。TCP-Reno では、3 回の重複確認応答の後、輻輳ウィンドウサイズを 1/2 に落として、輻輳回避フェーズに移り、再送を行っている。この手法であると、例えば図 2.5 のように連続して、パケットが損失している場合に効率が悪い。損失していると

分かっている場合でも、3 回の重複確認応答を待って、再送を行い、その度に輻輳ウィンドウサイズを半分まで下げるためである。そこで、TCP-NewReno では、重複確認応答を 3 回確認するまでに送信したパケットの ACK を受信するまでは、再送モードを継続するアルゴリズムになっている。この手法であれば、パケット損失が連続した場合でも、TCP-Reno に比べ、転送効率の下げ幅は少なくなる。
図 2.5 は TCP-NewReno による再送の様子を示している。

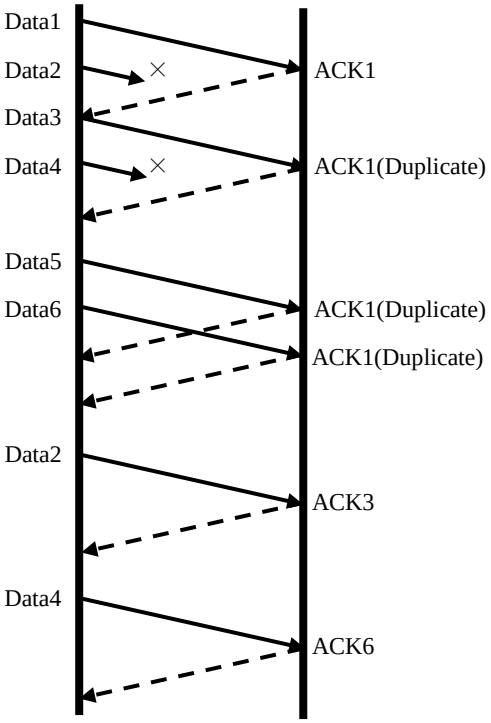


図 2.5 TCP-NewReno による再送の様子

図 2.5 ではまず、データ 2 が損失しているために、ACK1 を重複して受信している。そのため、重複応答を 3 回受信した後、送信側はデータ 2 を再送する。この場合、データ 6 までを送信しているので、ACK6 が返ってくるべきなのだが、返ってくる ACK は ACK3 である。TCP-NewReno ではこの場合、データ 4 も損失していると考え、重複応答を 3 回待たずに、データ 4 も再送する。再送して ACK6 を受信した後は、また Reno と同様のアルゴリズムに移る。

2.2.3 SACK[11]

SACK とは TCP-Reno に選択的再送(Selective ACK)を加えたもので Reno with SACK option と呼ばれている。図 2.6 に Selective ACK の様子を示す。

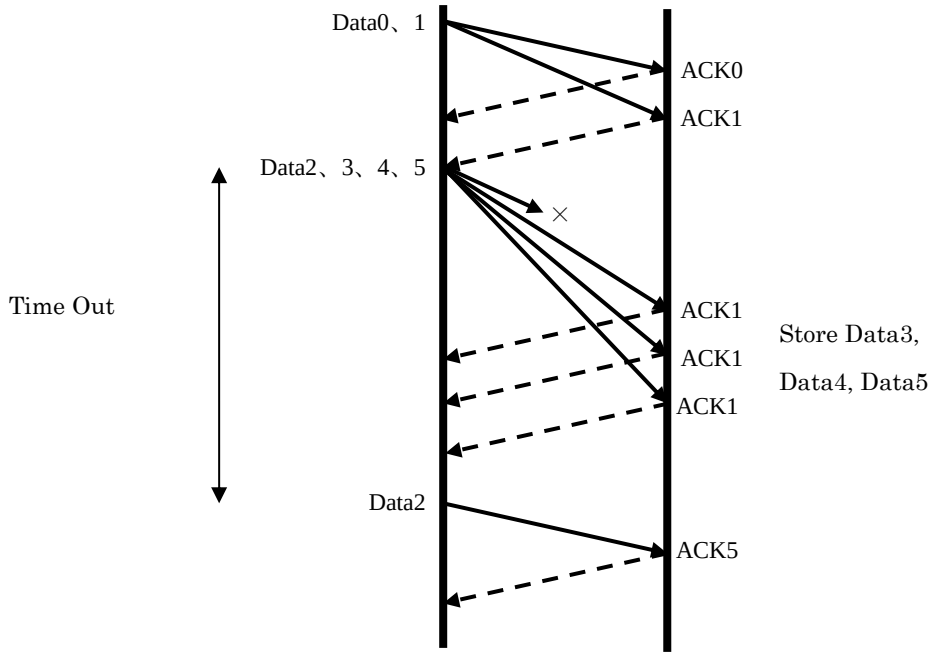


図 2.6 選択的再送の様子

図 2.6 では、データ 2 が損失しているため、ACK1 を重複して受信している。この際、受信側はデータ 2 以降の正しく受信できたデータを廃棄せずに受信バッファに保存する。そして、送信側からデータ 2 が再送された際には、正しく受信された最後のデータを知らせる ACK(ここでは ACK5)を送信する。このようにして、正しく受信できたデータを廃棄しない事で、無駄な再送を減らしているのが Selective ACK であり、それを加えたものが、SACK である。

第 3 章　ストリーミング

音声や動画などのマルチメディアデータを転送・再生する方法は、ダウンロード方式とストリーミング方式に大きく分けられる。ダウンロード方式は全てのマルチメディアデータをダウンロードしてから、再生を行い、ストリーミング方式はマルチメディアデータをダウンロードすると同時に再生を行う。そのため、ダウンロード方式に比べると待ち時間を大幅に減らすことが可能になり、またライブ中継を行う事も可能になるという利点がある。ストリーミングを配信するプロトコルには、UDP が利用される。UDP は TCP と異なり、再送制御等で信頼性を保証するわけでもなく、輻輳制御をすることはないが、一定の送信レートでパケットを送り続けるため、常に一定の帯域を必要とするリアルタイム通信に向いているとされているためである。しかし、UDP のみでは、ストリーミングを実現する事は難しく、アプリケーション層の RTP/RTCP を合わせて使用する必要がある。以下で、UDP ではカバーしきれないストリーミングの課題と RTP/RTCP、RTSP について述べる。

3.1　ストリーミングの課題

ストリーミングの課題として、大きく同期再生・パケット廃棄対策、輻輳制御があげられる。前述の通り、UDP はこれらの対策をしないため、RTP/RTCP がカバーする事になる。

3.1.1　同期再生

到着時間がばらつくパケットからどのように同期再生を行うかという問題がある。大きく、メディア内同期、メディア間同期、システム同期が存在する。

3.1.1.1　メディア内同期

メディア内同期とは、音声のみ、もしくはビデオのみ等、単一メディアの同期再生の事を言う。送信側から送られるパケットを、受信側は一定のタイミングで受信できるわけでなく、ネットワークの状況などから、必ず、到着時間の揺らぎ(ジッタ)が存在する。そのために、到着するデータの順番が乱れ、それに伴う音声の途切れなどが出る場合などがある。放送などの片方向通信であれば大容量のバッファにデータを溜め、この問題を解決することができるが、品質要求の厳しい通信などの双方向通信では、用意できるバッファは小容量であり、許されるバッファリング遅延は少なく、問題の解決は難しい。そこで、RTP のタイムスタンプが利用されている。

3.1.1.2　メディア間同期

メディア間同期は、音声とビデオの同期等、複数メディアの同期再生の事を指す。音声は音声で、タイムスタンプを利用して、メディア内同期を行うが、ビデオはビデオ側でタイムスタンプを利用してメディア内同期を行う。その際に、音声と映像のずれが発生してしまう可能性がある。この問題を解決すべく、RTCP では共通(NTP)時間軸を利用して、音声やビデオのデータ再生時間に補正をかけ、メディア間の同期を図り、ずれをなくしている。

3.1.1.3　システム同期

複数端末の時刻合わせを行って、複数端末間での同期再生を指す。必須ではないが、共通(NTP)時間軸を利用して、複数端末の時刻合わせを行うことで、実現することができる。

3.1.2　パケット廃棄対策

UDP はパケット損失した場合などに、再送などの制御を行わず、常に一定の送信レートでパケットを送り続ける。高速な通信を実現するためには必要なことであるが、マルチメディアデータを受信して、再生する場合に、パケット損失が多いと、受信した映像がなんの映像かすらわからない場合がある。そのため、パケットを損失した場合に、損失したパケットを復号できる FEC パケットの制御や、パケット損失により、エラーが発生したフレームに対し、過去のフレームなどを利用してフレームを補完するエラーコンシールメントなど様々な対策が導入されている。

3.1.3　輻輳制御

UDP は輻輳制御を行わないため、共存する TCP フローを追い出してしまい、協調性の面で問題がある。そのため、TCP-Reno に等価なレート制御を行う TFRC(TCP Friendly Rate Control)が提案されている。TFRC は基本的に送信レートを変えるだけで、再送を行うことはなく、UDP の特徴を生かしながら、TCP との親和性を改善する事ができる。

3.2　RTP と RTCP

前述の通り、UDP のみでは、ストリーミングの課題を解決する事は難しい。そのため、上位層の RTP/RTCP を用いて、同期再生、パケット廃棄対策を行っている。RTP/RTCP はリアルタイムアプリケーション特有の層であるアダプテーション層で実装されている。

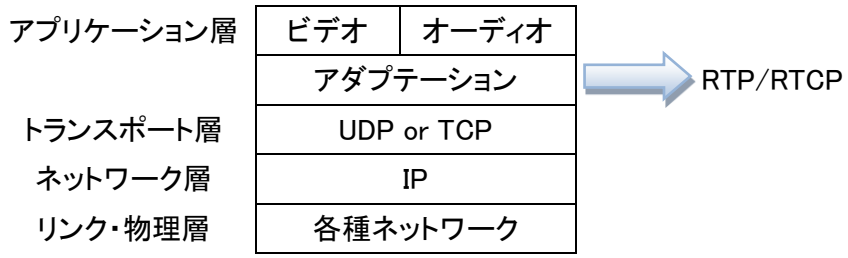


図 3.1 アダプテーション層の様子([14]より)

3.2.1　RTP (Realtime Transport Protocol)[12]

音声や映像をストリーミング再生するためのプロトコルで、パケットにシーケンス番号や、タイムスタンプを付加する事によって、再生するタイミングを調整する事ができる。しかし、QoS を保証するというわけではなく、通常、RTCP とセットで用いられる。以下に RTP ヘッダの構造と RTP の特徴を示す。

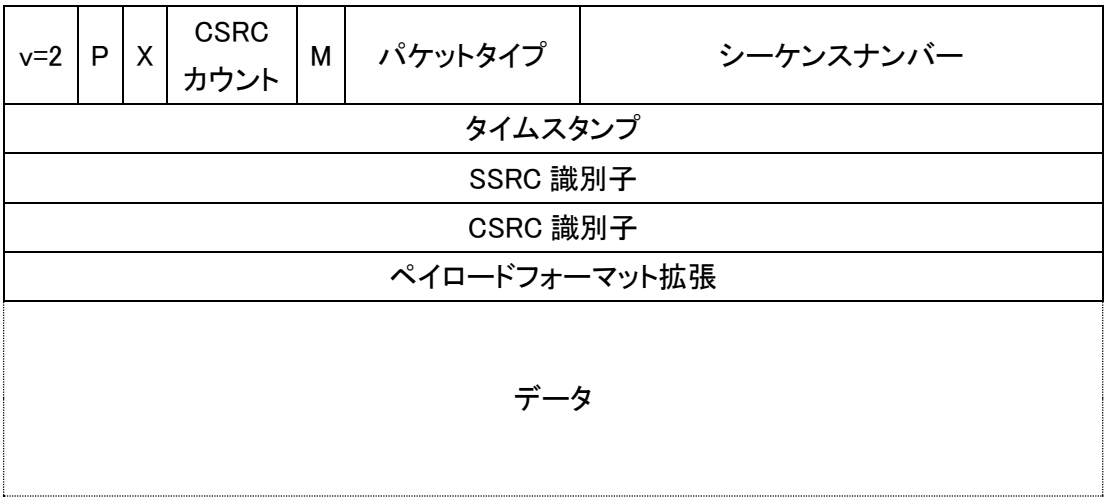


図 3.2 RTP ヘッダの構造([14]より)

v:RTP のバージョン識別子である。通常、バージョンは 2 である。

P:パッディング設定フラグであり、P=1 の時、RTP パケットの終端にパッディングオクテットが追加されている。

X:拡張ヘッダを設定するフラグであり、X=1 の時は、RTP ヘッダの後に拡張ヘッダが追加される。

M:パケットの区切りを示すものである。音声の場合、有音区間と無音区間の区切りを、映像の場合、フレームの区切りを示す。M=0 の間は、有音区間、もしくは同一フレーム区間であり、M=1 の時、有音区間の終了やそのフレームの終了を示す。

パケットタイプ(PT):メディアフォーマットの通知である。例えば PT=26 であればメディアフォーマットは JPEG であることを示し、PT=33 であれば、MPEG2-TS である事を示す。

シーケンスナンバー:RTP がパケットに付加する番号で、パケット廃棄対策として利用される。シーケンスナンバー番号の不連続を検出した場合、パケットが損失していると判断でき、めったに発生しないが、シーケンス番号の順番が逆になっていたら、パケットの到着順序の逆転を検出する事ができる。

タイムスタンプ:パケットの再生時間が示されている。メディア内同期に利用され、到着時間が揺らいだとしても、RTP タイムスタンプを利用する事で、再生時刻の復元が可能となる。

SSRC 識別子:同一の RTP セッション内の参加者の識別子。ランダムに付与され、他の参加者とかぶらないようにしなければならない。

CSRC 識別子:データを識別するために CC フィールドから付与される識別子。

ペイロードフォーマット拡張:ペイロードタイプによって、任意に拡張される。データに再同期情報を格納した拡張ヘッダを付加することで、パケット損失が発生した場合でも、パケットを復号することが可能になる。

3.2.2 RTCP (RTP Control Protocol)[12]

RTP と組み合わせて用いられるプロトコルで、RTP でデータを送受信するためのセッションを制御するプロトコルである。定期的に RTCP-RR(Receiver Report)、RTCP-SR(Sender Report)を送受信し、定期的に伝送遅延やパケット廃棄状況を得ることができる。また、RTCP では、NTP タイムスタンプを利用して、メディア間同期を行っている。以下に RTCP ヘッダの構造を示す。

v=2	P	RC	PT=SR=200	パケット長
送信元 SSRC 識別子				

図 3.3 RTCP 共通ヘッダ([14]より)

NTP タイムスタンプ(MSB)
NTP タイムスタンプ(LSB)
RTP タイムスタンプ
送出パケット数
送出バイト数

図 3.4 RTCP-Sender Report の構造([14]より)

SSRC 識別子 #n	
瞬時廃棄率	累積廃棄パケット数
シーケンスナンバーの最大値	
ジッタ遅延	
最新の SR 受信時の NTP タイムスタンプ(LSR)	
LSR から現在までの遅延(DLSR)	

図 3.5 n 番目の RTCP-Receiver Report の構造([14]より)

3.3 RTSP (Real Time Streaming Protocol)[13]

音声や動画などのデータをリアルタイムに配信するアプリケーション層のプロトコルで、エンドユーザー間のセッションの設立や、コントロールを行う。以下に RTSP の特徴を示す。

3.3.1 RTSP Method

RTSP でセッションの設立やコントロールを行う際に、サーバー・クライアント間でやりとりされるメッセージが RTSP Method である。図 3.6 は Method の種類を記述したものであり、図 3.7 は RTSP Method のやりとりによって、セッションの開始、停止・再開・終了の様子を示したものである。

メソッド	方向	要求条件	内容
DESCRIBE	C→S	推奨	セッション情報の取得
ANNOUNCE	C→S S→C	オプショナル	C→S:クライアントからの セッション情報の通知 S→C:セッション情報の更新
GET_PARAMETER	C→S S→C	オプショナル	セッションパラメータの取得
OPTIONS	C→S S→C	必須	オプション機能のチェック
PAUSE	C→S	推奨	メディア転送の中断
PLAY	C→S	必須	メディア転送の開始、再開
RECORD	C→S	オプショナル	メディア情報の記録
REDIRECT	S→C	オプショナル	リダイレクション
SETUP	C→S	必須	セッションの初期化
SET_PARAMETER	C→S S→C	オプショナル	セッションパラメータの設定
TEARDOWN	C→S	必須	セッションの終了

図 3.6 RTSP Method([14]より)

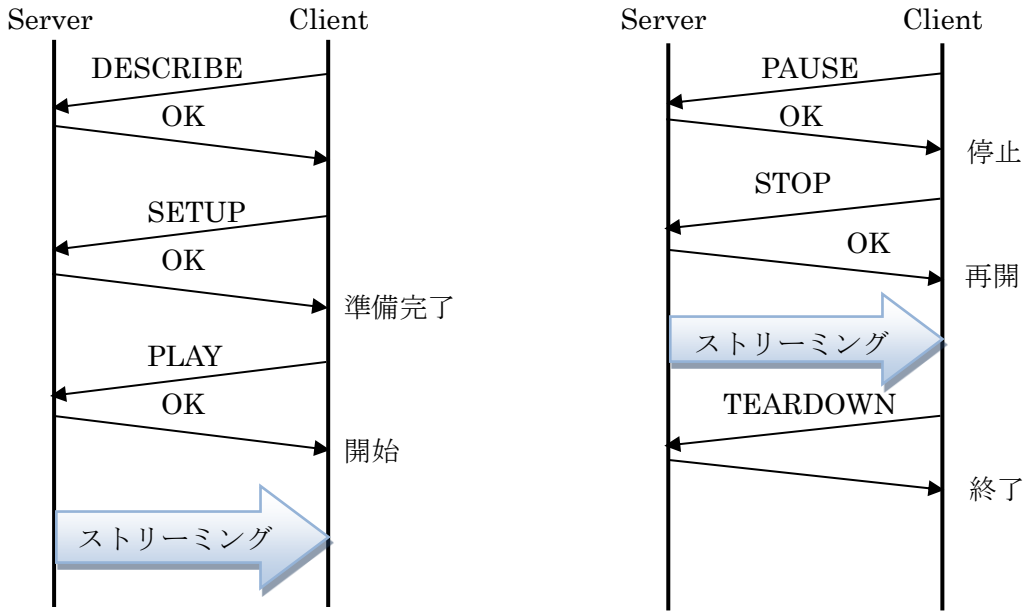


図 3.7 RTSP によるセッションの開始(右)と停止・再開・終了([14]より)

3.4 HTTP Live Streaming[3]

HTTP Live Streaming は、Apple 社が提案した、HTTP を利用してストリーミングデータをライブ配信する技術である。かつては、UDP/RTP と RTSP を用いて、ストリーミング配信を行うのが、主流の方式で HTTP ストリーミングはファイアウォールを超えるための手段として、使われていた。RTSP や UDP はファイアウォールを超えられない場合があるためである。しかし、近年では、広帯域化の進展とともに、Youtube に代表される HTTP Streaming が主流となりつつある。さらに、Apple 社の iPhone や iPad の普及を背景に HTTP Live Streaming は注目を集めている。

3.4.1 HTTP Live Streaming の仕組み

HTTP Live Streaming を行うためには、サーバー側で配信したい動画ファイルをエンコードし、分割を行い、同時にどの順番でファイルを再生すべきかを示すプレイリストファイルを作成する必要がある。[3]の実装では、ffmpeg コーデックを使用して、配信する動画を H264/AAC 形式にエンコードし、MPEG2-TS 形式のファイルに変換し、segmenter を利用して指定された秒数ごとにファイルを分割し、同時にプレイリストファイルを作成している。ユーザー側はプレイリストファイルに従って、分割されたファイルを順番に HTTP でダウンロードする事によって、動画の再生を行う事が出来る。プレイリストファイルは図のようなシンプルなもので、動画の再生は EXT-X-ENDLIST まで行われる。ただし、再生するユーザー側には Mac OS X や iPhone OS 3.0 もしくは iPad、iPod touch に加え、Quick Time x が必要となる。

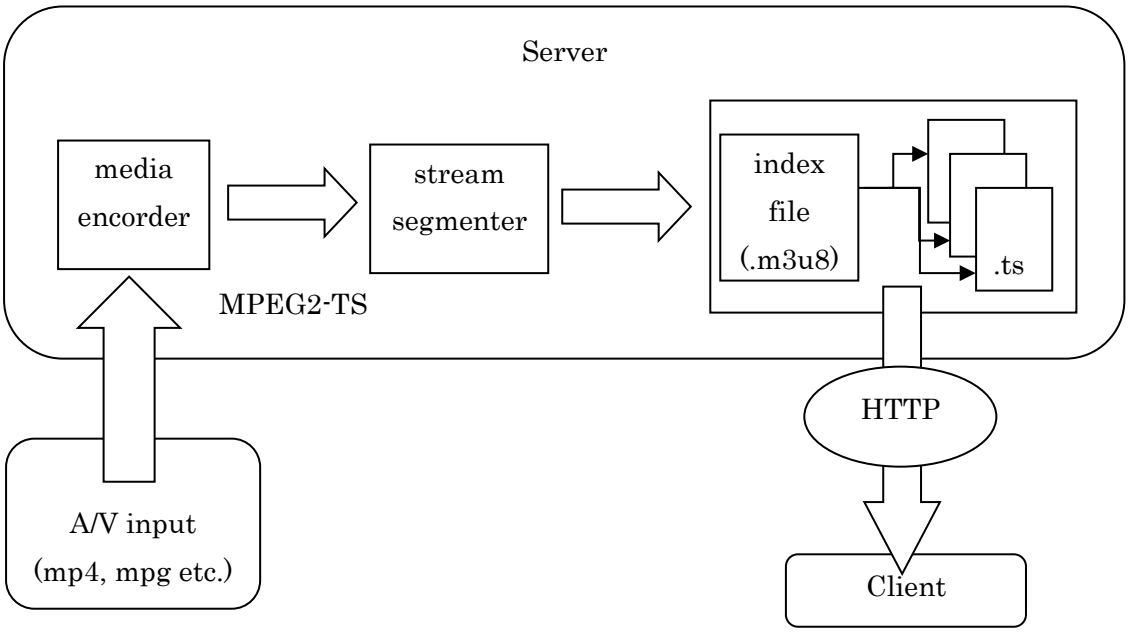


図 3.8 HTTP Live Streaming([3]より)

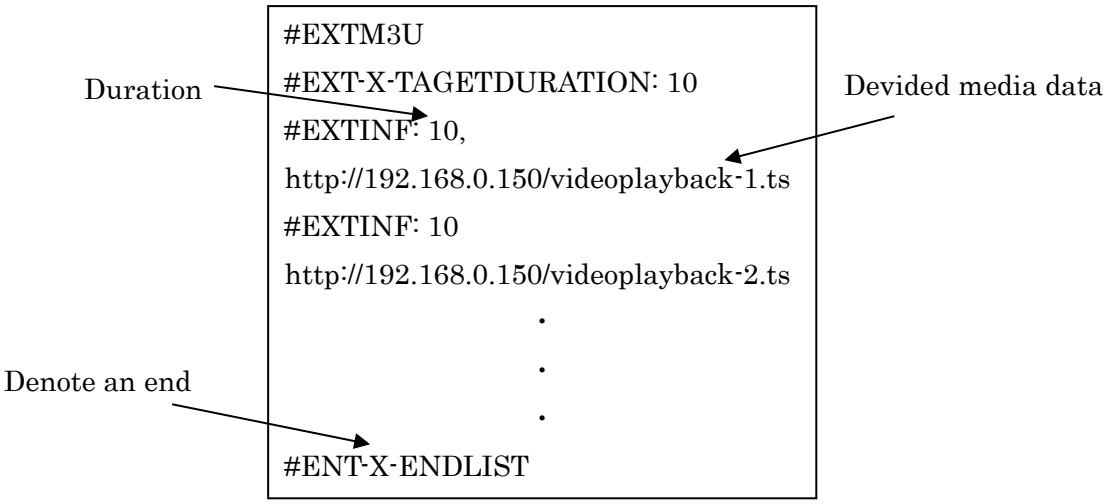


図 3.9 プレイリストファイル

3.4.2 HTTP Live Streaming の再生

前述の通り、HTTP Live Streaming を再生する際、クライアント側がプレイリストファイル(.m3u8)を読み込み、プレイリストファイルに書いてある順番で分割されたファイルにアクセスし、再生を行う。図 3.9 のようなプレイリストファイルであれば、まず、一番始めにプレイリストファイルを読み込み、次に、videoplayback-1.ts にアクセスして再生、その次は videoplayback-2.ts にアクセスして再生というような手順で動画を再生していく。そして、最後に再び、プレイリストファイルを読み込んで再生終了となる。そのため、通常のストリーミングの再生では、コネクションを確立したら再生終了までコネクションを切断する事はないが、HTTP Live Streaming では、違うファイルにアクセスする度にコネクションの確立・切断を繰り返すことになる。図 3.10 は HTTP Live Streaming における、コネクションの確立・切断の様子を示したものである。

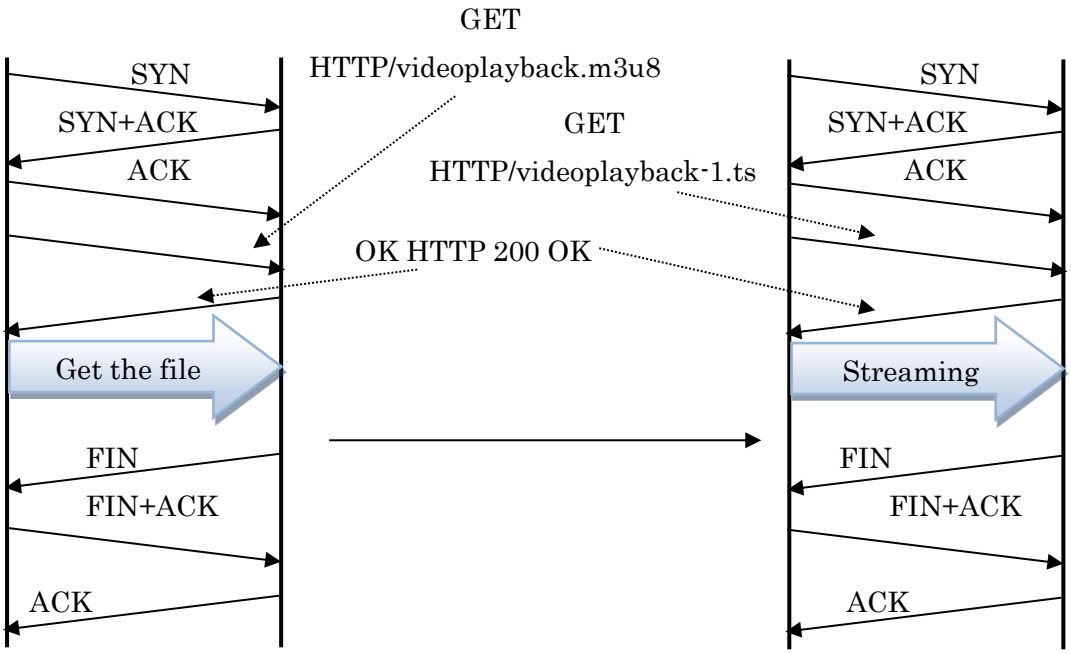


図 3.10 HTTP Live Streaming における再生の様子

3 ハンドシェイクでコネクションを確立し、ストリーミング再生をする。再生が終了したら、コネクションの切断を行い、また、つぎのファイルを再生する際は 3 ハンドシェイクでコネクションを確立し直している事がわかる。

3.4.3 HTTP Live Streaming の利点

3.4.3.1 QoS の保証

HTTP Live Streaming では、異なる帯域幅の動画を用意し、インデックスファイルにその情報を記述することができる。これにより、動画を再生するクライアント側では、自身の環境に応じて、帯域幅の異なる動画に切り替える事が可能である。例えば、自身のネットワークの環境が悪くなった時などは、今よりも低い帯域幅の動画に切り替える事が可能である。また、異なる帯域幅の情報だけでなく、動画を配信するサーバーが 2 つあるならば、その情報をインデックスルファイルに記述することもできる。例えば、サーバーがクラッシュした場合などで、動画の再生をすることができなくなっても、クライアントはインデックスファイルに記述してある別のサーバーにアクセスする事で、動画の再生が可能になる。

3.4.3.2 セキュリティ

HTTP Live Streaming では、ファイルを暗号化することができる。サーバー側で、動画ファイルをキーで暗号化し、そのキー情報をインデックスファイルに記述する。クライアント側は、その暗号化キーを使いながら再生を行っている。

3.5　HTTP Dynamic Streaming[4]

HTTP Dynamic Streaming は Adobe が提案している HTTP ストリーミングでビデオオンデマンドにも、ライブ配信にも対応している。再生には、Adobe Flash Player 10.1、もしくは Adobe AIR 2 が必要である。

3.5.1　HTTP Dynamic Streaming の仕組み[15]

HTTP Dynamic Streaming を配信するためには、HTTP Live Streaming と同様に、サーバー側での準備が必要である。サーバーで、配信する動画を VP6/MP3、または、H.264/AAC 形式にエンコードし、分割し、F4F 形式のファイルを作る。F4F ファイルを作るのと同時に、動画情報を格納したマニフェストファイルを作成する。ユーザー側は、再生フレームワークである OSMF(Open Source Media Framework)を利用して、このマニフェストファイルを読み込んで、再生すべきファイルをダウンロードし、動画の再生を行う。図 3.11 は HTTP Dynamic Streaming の仕組みを示したものである。

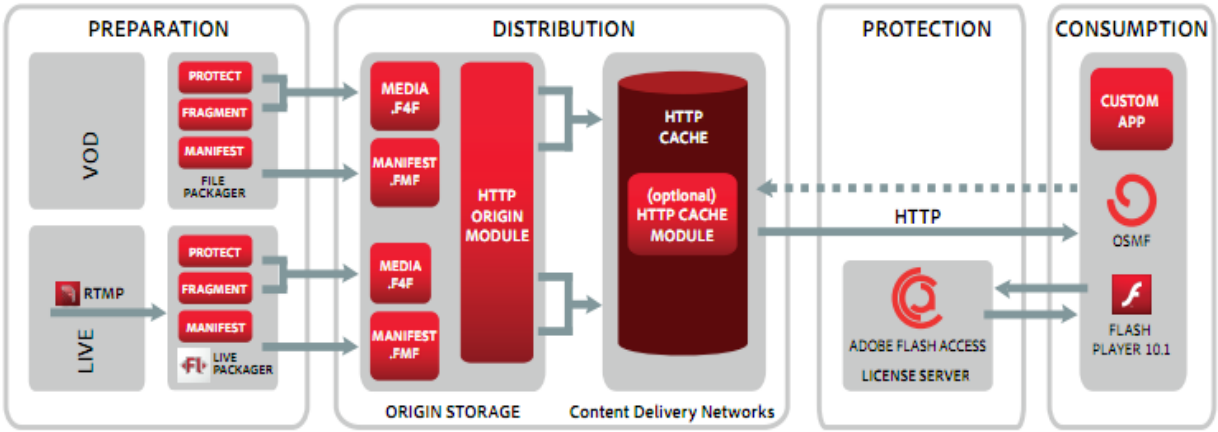


図 3.11 HTTP Dynamic Streaming workflow(出典:参考文献[15])

3.5.2　HTTP Dynamic Streaming の利点[15],[16]

HTTP Dynamic Streaming の利点として、以下のような点があげられる。

3.5.2.1　高い互換性

Adobe Flash Player 10.1、もしくは Adobe AIR 2 がインストールされていれば、Windows、Mac OS、Linux など様々な OS で、HTTP Dynamic Streaming を再生する事ができる。

3.5.2.2　DVR 機能とセキュリティ

例え、Live 配信であっても、停止・巻き戻しといった操作を実現する事ができる。また、Adobe Flash Access 2 によって、オンデマンド・ライブ配信を問わず、ファイルを暗号化して、安全な動画再生を行うことができる。

3.5.2.3　高い QoS 保証

様々なビットレートの動画を用意し、マニフェストファイルに動画に付随しているビットレートを記述している。これにより、OSMF は再生側の処理能力や、帯域幅に応じて、

ビットレートの切り替えを行う事で、ユーザーの環境に最適な動画を提供する事ができる。

3.6 DASH(Dynamic Adaptive Streaming over HTTP)[7]

前述の通り、HTTP Live Streaming や Dynamic HTTP Streaming といった技術は、HTTP を利用しているので、CDN や HTTP キャッシュが活用でき、ファイアウォール等も容易に越えられるという利点がある。ただ、これらの技術はユーザーに要求されたビットレートのストリーミングデータをネットワークの状況に関係なく、送信しようとする。そのため、ネットワーク環境が悪ければ、動画の停止や再生が途切れるといった状況が発生してします。これに対して、DASH はネットワークの環境に応じて、動的にストリーミングデータのビットレート等を変化させるため、その環境に適した動画の配信を行う事ができる。例えば、ネットワークの環境が悪化した場合には、ストリーミングデータのビットレートを下げる事で、動画が途中で停止してしまう事や、再生が途切れるといった状況を未然に防ぐことができる。また、逆にネットワークに余裕がある場合には、ストリーミングデータのビットレートを上げる事で、ユーザーにより高品質な動画を提供する事ができる。

3.6.1 DASH の仕組み

基本的な仕組みは、HTTP Live Streaming と同じである。サーバー側で配信したい動画ファイルをエンコードし、分割を行う。ユーザーはプレイリストファイル(MPD)に従って、その分割ファイルを順番にファイルをダウンロードする事で、動画の再生を行う。この際、ユーザーが状況に応じて、動画の品質を変化させるために、サーバー側では様々なタイプのストリーミングデータを用意していなければならない。例えば、ビットレートを変化させたストリーミングデータを用意していれば、ユーザーはネットワーク環境に応じて、動画のビットレートを変更する事で動画の再生が途切れないように試みることができる。また、動画の音声等を複数言語用意しておけば、ユーザーは自身の望む言語での動画の再生が可能となる。さらに動画の分割時間を変えることで、On-demand から Live のストリーミングまで幅広く対応する事ができる。このように、動画のタイプを動的にユーザー側で柔軟に変更できる点が DASH の魅力だが、ユーザーにどのようなタイプの動画がサーバーにあるかを知らせるために、前述の通り、MPD(Media Presentation Description)を作成する事が必要である。次節では、MPD について述べる。また図 3.12 は DASH の仕組みを示したものである。ユーザーはダウンロードした MPD ファイルを基に、DASH の再生を行う。

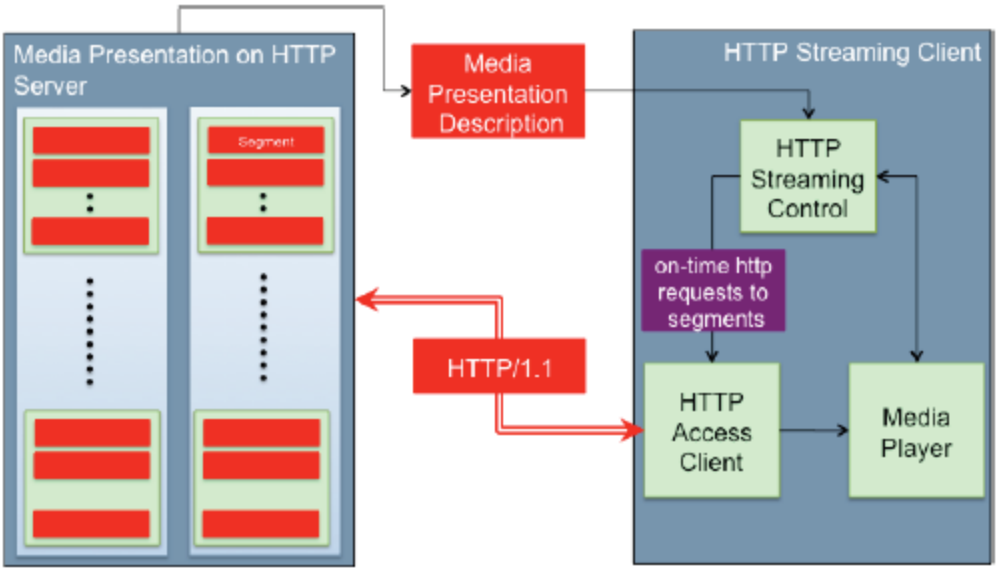


図 3.12 DASH の仕組み(出典:参考文献[7])

3.6.2　MPD について

MPD ファイルは XML で記述されており、構成は図 3.13 のようになっている。図からわかるように、MPD ファイルは大きく 4 つのレベルに分けることができる。以下に、各レベルが担う役割を簡単に記述する。

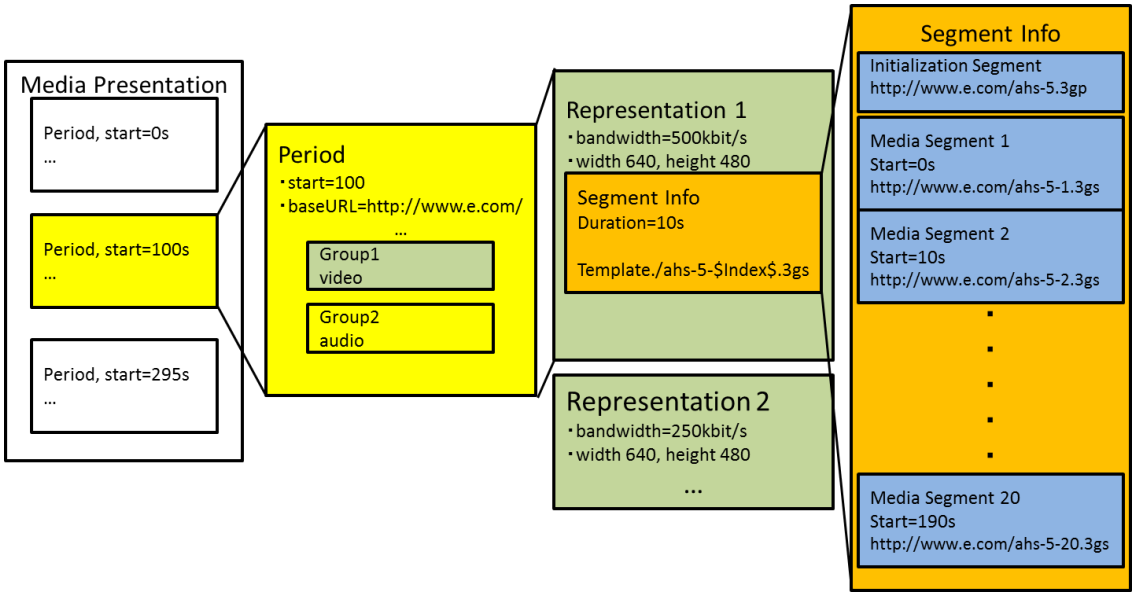


図 3.13 MPD ファイルの構成例([7]より)

Media Presentation : MPD のトップレベルであり、映画などの動画をエンコードして 1 つのビデオストリームとして構成したものと考えてよい。また、1 つ以上の Period から構成されている。

Period : Period はその 1 つのビデオストリームを分割したものと考えられ、動画の開始時

刻と、分割した動画 1 つの長さである Duration を示している。また、Period は、分割された動画の境界線とも考えられ、サーバーのある URL やエンコーディングのパラメータなども示している。また、Period は広告などの新しいコンテンツを挿入する場所としても扱われる。また、1 つ以上の Group から構成される。

Representation : Period 内の Group を構成する。Representation では、その動画のビットレート、解像度、音声や字幕の言語等の情報を示している。また、Representation は 1 つ以上の Segment Info を含み、この Segment Info に再生すべき動画の URL の情報が含まれている。

Segment Info : Information Segment と Media Segment から成る。Information Segment は、ストリーミングデータは含んでいないが、その Representation にアクセスするための情報を含んでいる。Media Segment はその動画のある URL を含んでいる。

DASH はライブストリーミングにも対応する事ができるが、その場合は、新たなストリーミングデータの Segment や Period が作られるたびに、MPD ファイルを更新し、ユーザー側に更新された MPD ファイルを送る必要がある。また、ユーザー側とサーバー側間で、きちんと同期がとれている事を保証するために、MPD で記述される時間は UTC(Universal Time Clock)に合わせる必要がある。

3.6.3 DASH の再生

図 3.14 を参考に、どのように DASH の再生が行われるかを簡潔に述べる。なお、図 3.14 は On-demand の場合の再生の流れを示したものである。また、URL に rep-192 を含む動画はビットレートが 192[kbps]、分割時間が 80[s]のタイプで、ここでは Representation1 と表記する。また、rep-384 を含む動画はビットレートが 384[kbps]、分割時間が 40[s]のタイプで Representation2 と表記する。

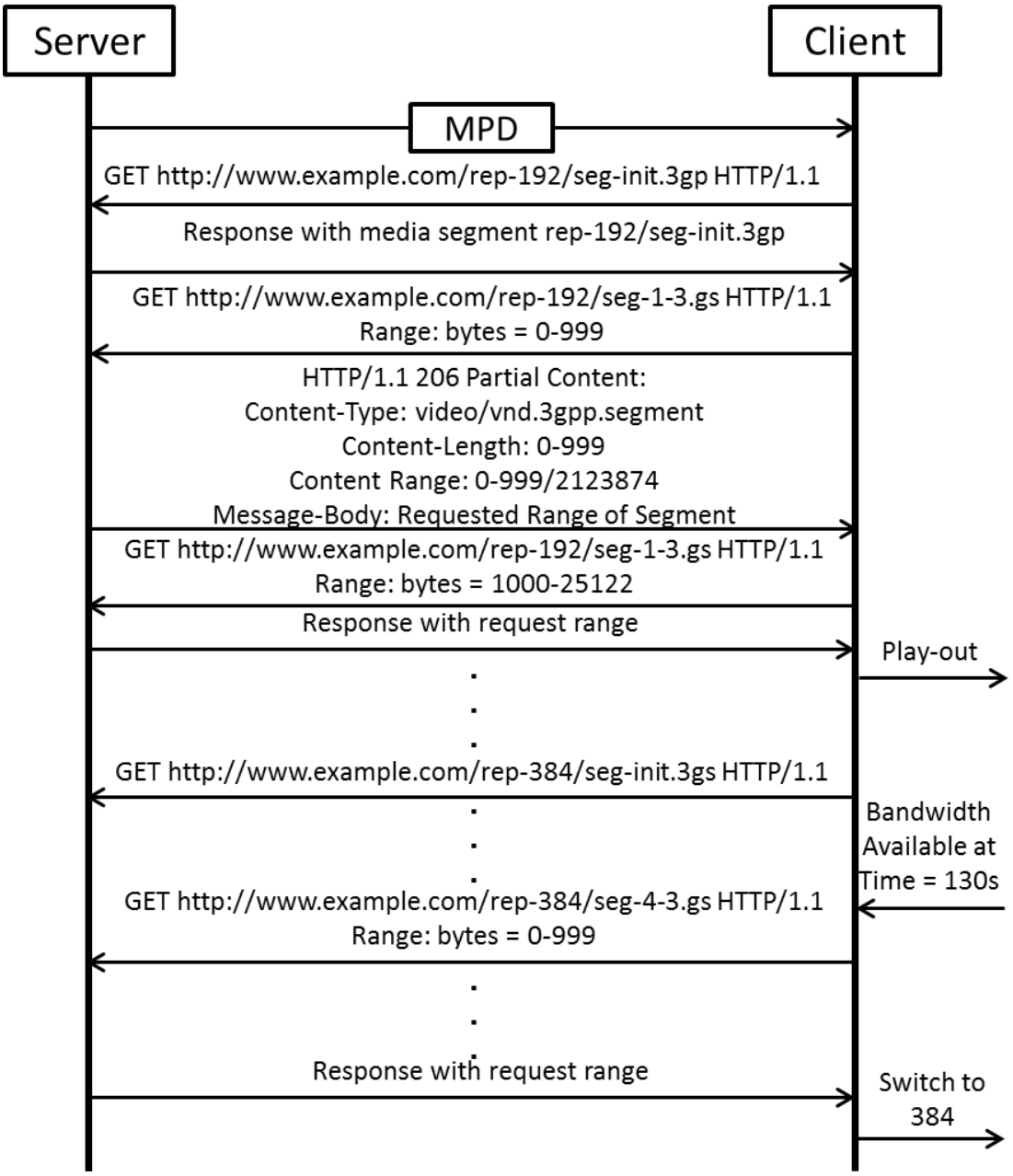


図 3.14 DASH による動画再生の流れ([7]より)

1) クライアントが MPD ファイルをサーバーからダウンロードし、その情報を基に、Representation1 の動画の Initial Segment をダウンロードする。

2) Representation1 の最初の Media Segment を要求する。

3) 動画の再生が開始される。

4) クライアントが Representation1 の再生と並行して、違う Representation の動画(ここでは Representation2)への切り替えの準備をする。そのために、Representation2 の Initial Segment のダウンロードを開始する。

5) クライアント側の予測帯域が Representation2 を再生に十分な値になったので、(ここ

では再生開始から 130[s]にその状態になったとする)Representation2 の Media Segment をダウンロードする。

6) Representation2 の動画の再生が開始され、Representation2 の動画への切り替えが完了する。

3.6.4 Adaptive Streaming における関連研究について

Adaptive Streaming に関する関連研究について述べる

関連研究として、DASH の改善がある。DASH に CMP(Composition of Media Presentation)という概念を取り入れ、ユーザー側で動画再生時に選択できるオプションを増やす手法[17]や、QoS の要素だけでなく、QoE の要素も取り入れて動画レベルの切り替えを行う QDASH という手法[18]が提案されている。また、SVC(Scalable Video Coding)を取り入れることで、サーバーやネットワークの負荷を軽減し、キャッシュヒット率を高めるといった効果が期待できる iDASH[19]も提案されている。

また、動的なレート制御を行うアーキテクチャについても様々な関連研究がある。QAC(Quality Adaptation Controller)というものをサーバーに用意して、レート制御を行う手法[20]や、SPONGE(Stream Pooler Over a Network Graded Environment)[21]や、bandwidth manager[22]がある。[21]は、有線環境と無線環境が混じる実際のネットワーク形態に近い環境でスループット等の QoS のパラメータに関し高い効果を得られる事、[22]はホームゲートウェイに設置する事で、QoE に関して高い効果が得られる事が分かっている。

既存の Adaptive な動画再生プレーヤーや、動画再生技術の性能評価を行っている研究もある[23][24]。[23]は Smooth Streaming Player や、Netflix Player、Adobe OSMF Player の性能比較を行っており、[24]は車両で移動するという環境下で MSS、Adobe HTTP Dynamic Streaming、Apple 社の HTTP Live Streaming、DASH について比較実験を行っている。

第 4 章　LTE(Long Term Evolution)

4.1　LTE とは[25][26][27]

LTE とは従来の 3G で使われていた CDMA2000 や、3.5G で使われていた HSPDA といった技術を発展させた無線技術の事であり、2010 年 12 月 24 日から、NTT DOCOMO 社より LTE サービス「Xi」が開始されている。従来の技術に比べて、高速・大容量で低遅延での無線通信が期待されている。表 4.1 は 3G で使われていた W-CDMA や 3.5G で使われていた HSPA と、LTE を簡単に比較したものである。なお、例えば周波数使用効率が 3 倍という事は、ユーザーの通信量が 3 倍に増えたとしても、今までと同じ周波数帯域幅で、かつ、同じセクタで同じ数のユーザーを収容できるという事を示している。

表 4.1 W-CDMA/HSPA と LTE の比較([25]より)

	W-CDMA/HSPA	LTE
最大通信速度	下り:14.4Mbps 上り:5.76Mbps	下り:326.4Mbps 上り:86.4Mbps
遅延時間 (目標値)	なし	接続遅延 100ms 伝送遅延 5ms
周波数帯域幅	5MHz(固定)	1.4MHz,3MHz,5MHz, 10MHz,20MHz(可変)
周波数利用効率	上り:1 倍 下り:1 倍	上り:2 倍以上 下り:3 倍以上
通信方式	回線交換 (音声通話用) パケット交換 (データ通信用)	パケット交換

なお、LTE での最大通信速度が 326.4[Mbps]となっているが、この速度は帯域幅が 20[MHz]で 4×4MIMO を利用した場合の値である。ただ、日本の場合、ユーザー数が既に膨大であり 20[MHz]すべてを LTE で利用する事は難しく、また、4×4MIMO に関してもアンテナの本数を増やすと電力量が増えるという状況があり、ただちにこの速度を実現するという事は難しい。表 4.2 は変調方式や MIMO、帯域幅の組み合わせによる通信速度の変化を示したものである。

表 4.2 LTE の通信速度([28]より)

		下り		上り	
変調方式		64QAM		16QAM	64QAM
MIMO		2×2	4×4	なし	
帯域幅	5MHz	42.5Mbps	80.3Mbps	14.4Mbps	21.6Mbps
	10MHz	85.7Mbps	161.9Mbps	28.8Mbps	43.2Mbps
	15MHz	128.9Mbps	243.5Mbps	43.2Mbps	64.8Mbps
	20MHz	172.1Mbps	325.1Mbps	57.6Mbps	86.4Mbps

次節以降では、LTE の高速遅延を実現するキーワードである、OFDMA や 64QAM といった技術や、LTE でのハンドオーバー手順について述べる。

4.2 LTE の高速通信を実現する技術

4.2.1 OFDMA(Orthogonal Frequency Division Multiple Access)[25][26][27]

OFDMA は LTE のダウンリンクで用いられている無線技術である。無線 LAN で利用される FDMA(Frequency Division Multiple Access)や 2G の PDU や GSM で利用されていた TDMA(Time Division Multiple Access)、3G で利用されていた CDMA(Code Division Multiple Access)とは異なり、直交する周波数軸と時間軸のチャネルを分割してユーザーにサブキャリアを割り振る方式である。そのため、複数のユーザーに、それぞれにとって最も伝送効率の良いサブキャリアを割り当てる事ができるので、ユーザーは無線状況に応じて良好なサブキャリアを使用できるので、フェージングに強く、高い周波数利用効率が期待できる。図 4.1 は各無線技術の比較したものである。

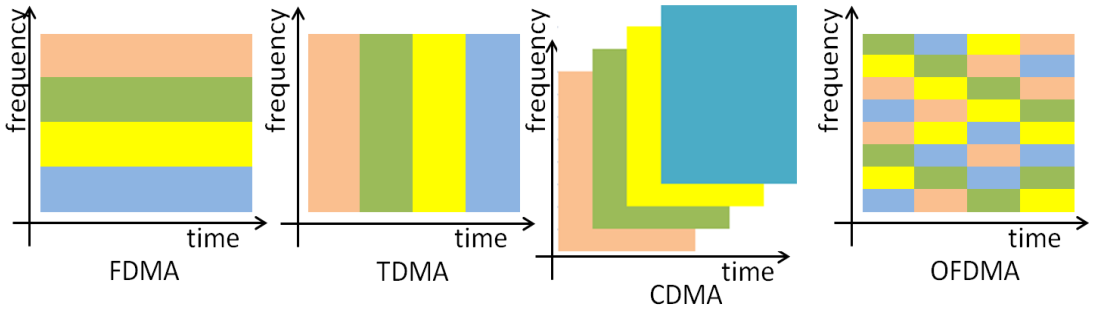


図 4.1 各無線技術の比較([27]より)

図 4.2 はサブフレーム構成とリソースブロック配置を示したものである。1 スロットは 7 つのシンボルからでき、また、15[kHz]間隔で隣接する 12 のサブキャリアで 1 つのリソースブロックが構成される。ユーザーはこのリソースブロックを割り当てられることでデータを送信する事が可能である。サブキャリアの数は帯域幅の値によって変化し、サブキャリアの数が多い程、高速での通信が可能となる。ただ、通常、周波数帯域幅は 10[MHz]のため、利用可能なサブキャリアは 600 サブキャリアとなり、1 サブフレームで送信可能なリソ

ースブロック数は 50 個となる。

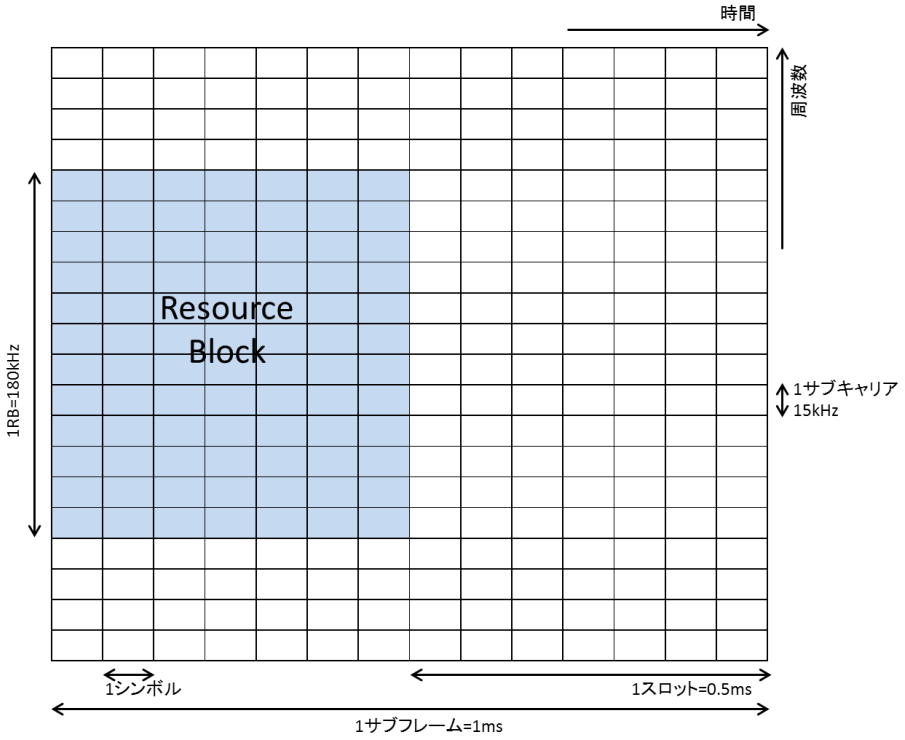


図 4.2 サブフレーム構成とリソースブロック配置([26]より)

4.2.2 LTE のチャネル構成

前述の通り、LTE の無線リソースの最小割り当て単位はリソースブロックだが、このリソースブロックに基地局がデータチャネル等を割り当てる。表 4.3 は物理チャネルと物理信号の概要を示したものである。LTE では表 4.3 のコントロールチャネルを基地局がスケジューリングして無線資源を割り当てている。このスケジューリングは各キャリアによって異なるとされ、スケジューラによって性能に差異が出る部分でもある。

表 4.3 物理チャネルおよび物理信号概要([26]より)

物理チャネル/物理信号名	用途
SS(Synchronization Signal)	セルサーチに用いられる同期信号
DLRS (Downlink Reference Signal)	下りリンクの伝送路推定、シンボルタイミング同期、受信品質測定、セル選択やハンドオーバーのための品質測定に使用される
PBCH (Physical Broadcast Channel)	UE がセルサーチ後に最初に読むべき最低限の情報(システム帯域幅、システムフレーム番号、送信アンテナ数)を送信するために使用される
PDSCH (Physical Downlink Shared Channel)	下りリンクのユーザデータを送信するための共有データチャネルである
PCFICH (Physical Control Format Indicator Channel)	各サブフレームの先頭の何個のシンボルが下りリンク制御情報を送信可能な領域として確保されているか通知するために使用
PHICH (Physical HARQ Indicator Channel)	PUSCH に対する ACK/NACK を送信するためのチャネルである
PDCCH (Physical Downlink Control Channel)	eNodeB がスケジューリングにより選択したユーザーに対して、無線リソースの割り当て情報を通知するために使用
DM RS (Demodulation Reference Signal)	上りリンクの伝送路推定、シンボルタイミング同期、受信品質測定などに使用される。
SRS (Sounding Reference Signal)	周波数スケジューリングを適用するために必要な受信品質測定やタイミング調整に使用される
PRACH (Physical Random Access Channel)	UE が初期アクセスやハンドオーバーにより、セルとコネクション確立を行う場合や再同期を行う場合に使用される
PUSCH (Physical Uplink Shared Channel)	上りリンクのユーザデータを送信するための共有データチャネルである
PUCCH (Physical Uplink Control Channel)	PDSCH に対する ACK/NACK や受信品質、スケジューリング割当要求信号を送信するために使用される

4.2.3 64QAM

OFDMA では、変調方式として 64QAM を採用している。64QAM は、電波と位相と振幅の 2 つを利用しており、4 つの位相を使って、1 シンボルで 4 ビットの信号を伝送する事ができる。そのため、図 4.3 からわかるように、QPSK と比べると、1 シンボルで送る事が

できる情報量が大きいため、通信速度が速くなる。従来、携帯電話などでは、QPSK 等が使われていたことを考えると、LTE が 64QAM を採用することで高速な通信を実現できている事がわかる。

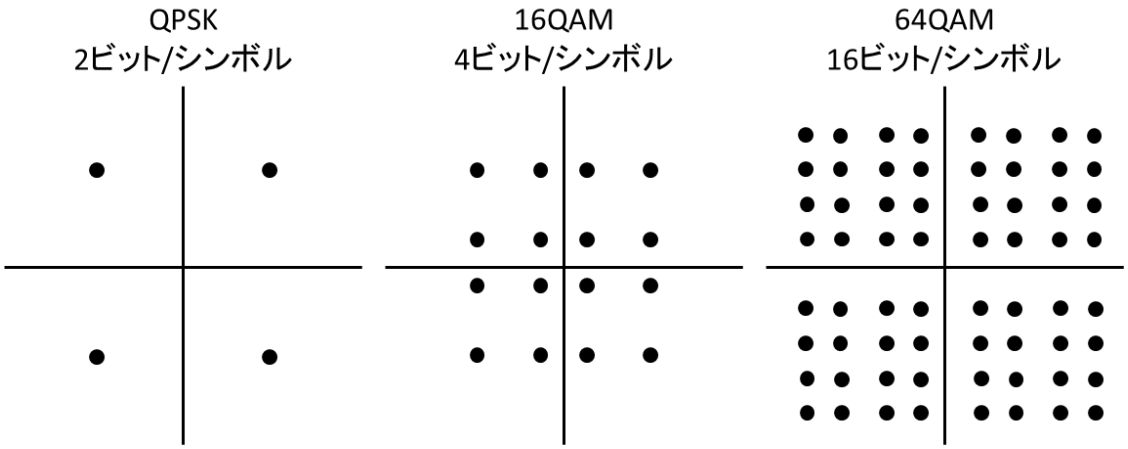


図 4.3 各変調方式の比較([27]より)

4.3 LTE におけるネットワーク構成[29]

LTE における基本的なネットワーク構成を図 4.4 に示し、各システムについて簡潔に述べる。

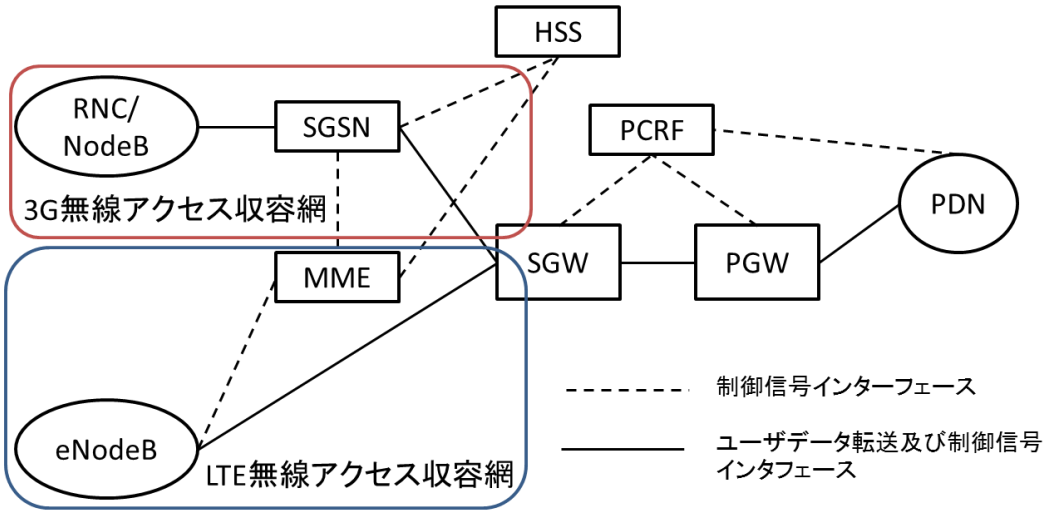


図 4.4 LTE における基本的なネットワーク構成([29]より)

eNodeB：LTE 端末を収容する基地局の事であり、ユーザーから最も近い構成要素である。

RNC/NodeB：NodeB は 3G 端末を収容する基地局であり、RNC (Radio Network Controller)は無線ネットワークにおいて基地局制御を行う。

SGW (Serving Gateway)：LTE 及び 3G ネットワークを収容し、ユーザデータの転送、IP アドレスの払い出し等を行う。そのため、異なる SGW へ接続しなおす場合は IP アドレスが変更される。

PGW (PDN Gateway) : PDN (Packet Data Network)とコアネットワークの接続点におかれるゲートウェイである。
PCRF (Policy and Charging Rules Function) : 課金制御ルールや QoS 制御を決定する装置で、SGW や PGW は PCRF の情報に基づき、パケット単位で制御を行う
MME (Mobility Management Entity) : LTE ネットワークにおいて、移動端末の移動管理や認証、ユーザデータの転送経路の設定を行う。実際にユーザデータ転送を行うのは前述の通り SGW で、MME ではユーザデータの転送は行われない。
HSS (Home Subscriber Server) : 加入者データのデータベースを管理する。
SGSN (Serving General Packet Radio Service Support Node) : 3G 無線アクセス収容網を構成する。

4.4 LTE におけるハンドオーバー手順[29][31]

LTE では、基地局同士が直接端末の情報をやり取りし、未達のパケットを転送する事ができる X2 ハンドオーバーを採用している。S1 ハンドオーバーに比べて、コアネットワークを介さないためコアネットワークの負担を減らすという利点があるが、X2 ハンドオーバーを行うためには基地局同士が同一の MME に接続されている必要性がある。そのため、異なる MME 間でのハンドオーバーでは S1 ハンドオーバーが利用される。図 4.5、図 4.6 にそれぞれのハンドオーバーの手順を示す。図 4.5 では、図 4.6 のように、SGSN や RNC を介することなく、未達パケットや端末情報の転送を基地局同士で行っており、コアネットワークの負担が減らされている事がわかる。

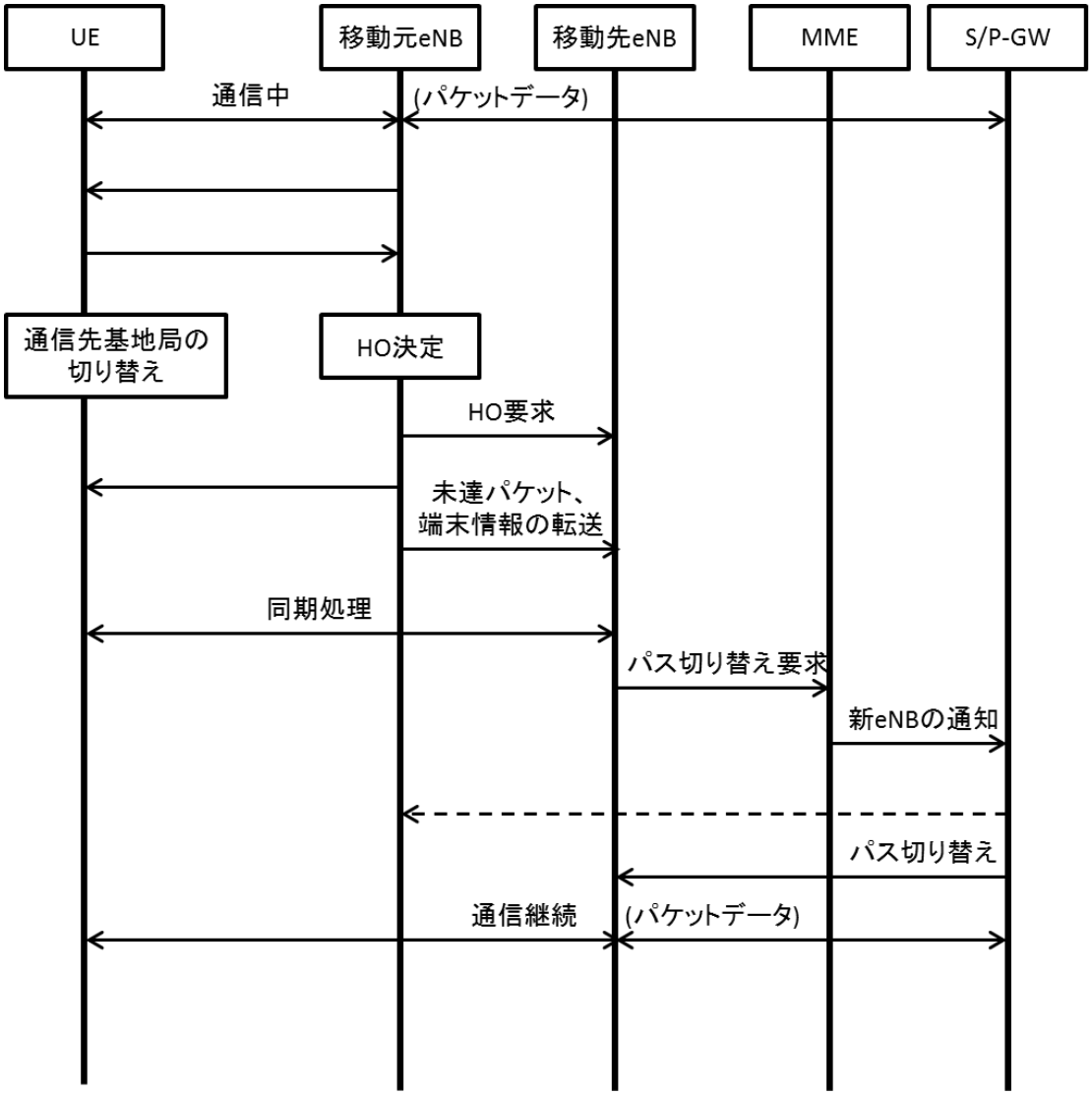


図 4.5 X2 ハンドオーバー手順([31]より)

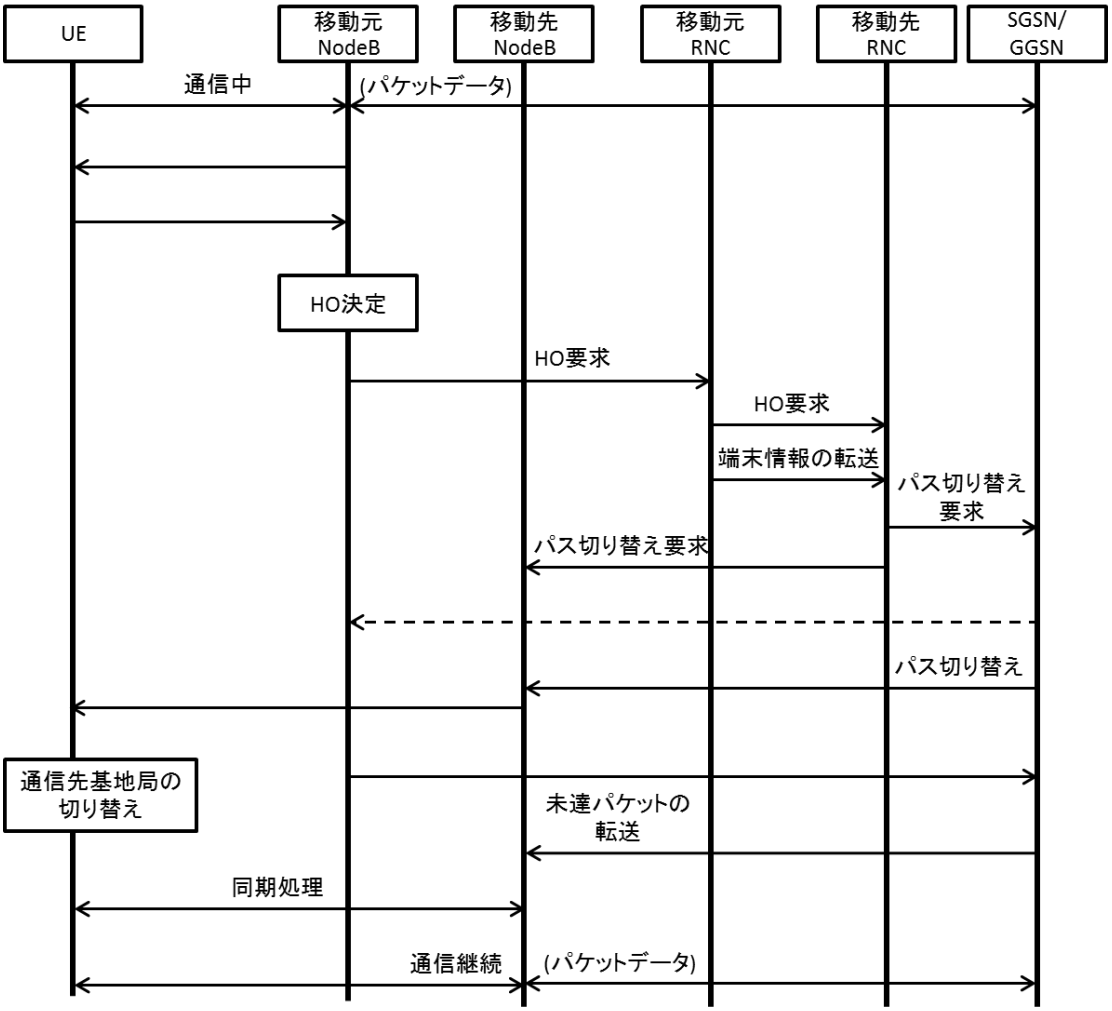


図 4.6 S1 ハンドオーバー手順([31]より)

4.5 LTE における実測の通信速度[30]

図 4.7 は株式会社 ICT 総研が全国 200 地点で、通信速度アプリ「RBB TODAY SPEED TEST」を利用して、LTE 通信速度を計測した結果をまとめたものである。なお、この調査では 1 つの測定地点において、下り通信速度、上り通信速度の計測をそれぞれ 3 回ずつ行った結果である。

（LTE受信地点数／測定地点数）　（実測値：Mbps）　（実測値：Mbps）

			LTE受信地点数			下り平均速度			上り平均速度		
			GALAXY SIII	iPhone5	iPhone5	GALAXY SIII	iPhone5	iPhone5	GALAXY SIII	iPhone5	iPhone5
			NTTドコモ Xi	au 4G LTE	ソフトバンク 4G LTE	NTTドコモ Xi	au 4G LTE	ソフトバンク 4G LTE	NTTドコモ Xi	au 4G LTE	ソフトバンク 4G LTE
1	首都圏エリア	25地点	19／25	21／25	**23／25**	8.59	12.71	**13.74**	1.76	5.84	**6.40**
2	名古屋エリア	25地点	20／25	18／25	**23／25**	7.91	8.43	**12.83**	1.31	5.09	**6.66**
3	関西エリア	25地点	15／25	17／25	**20／25**	5.06	8.51	**10.39**	1.19	4.15	**4.76**
4	札幌エリア	25地点	13／25	8／25	**22／25**	8.87	**11.67**	9.01	2.25	**5.46**	4.98
5	仙台エリア	25地点	15／25	20／25	**25／25**	5.34	6.72	**9.17**	0.84	2.88	**6.08**
6	広島エリア	25地点	14／25	16／25	**23／25**	7.35	8.34	**8.53**	1.57	4.00	**4.15**
7	高松エリア	25地点	19／25	15／25	**23／25**	**15.15**	11.21	11.05	3.40	4.96	**5.21**
8	福岡エリア	25地点	14／25	11／25	**21／25**	5.32	7.35	**11.62**	0.95	4.14	**4.97**
全国		200地点	129／200	126／200	**180／200**	7.95	9.37	**10.79**	1.66	4.57	**5.40**
平均速度トップ地点数						42地点	61地点	**97地点**	12地点	77地点	**112地点**

図 4.7 全国 200 地点通信速度測定結果(出典:参考文献[30])

4.1 節の表 4.2 に帯域幅、MIMO の組み合わせ等を変化させた場合の LTE の通信速度の値を示したが、直ちに表 4.2 のような速度の実現が難しい事は前述した。実際、図 4.7 から、キャリアごとに差はあるものの、スマートフォン LTE の実測の下り平均速度は、速くともだいたい 13[Mbps]前後であり、平均 9[Mbps]程度であり、実測の上り速度は速くともだいたい 5[Mbps]前後で、平均 4[Mbps]程度である事がわかる。また、LTE 受信地点数は NTT ドコモが、64.5%、au が 63%、ソフトバンクが 90%と急速に LTE カバーエリアが拡大している事もわかる。

第 5 章　シミュレーション実験

本章では、LTE でのハンドオーバーが発生する際の HTTP Live Streaming と DASH の性能評価を行った。分割ファイル 1 つのダウンロード時間とスループットを評価項目として、ハンドオーバーがストリーミング再生に与える影響を調査する。

5.1　実験環境

図 5.1、図 5.2 のようなネットワークトポロジーを用い、シミュレータとして Scenargie[32]を利用して評価実験を行う。図 5.1 はドメイン内ハンドオーバーを想定した実験トポロジーで、図 5.2 はドメイン間ハンドオーバーを想定した実験トポロジーである。

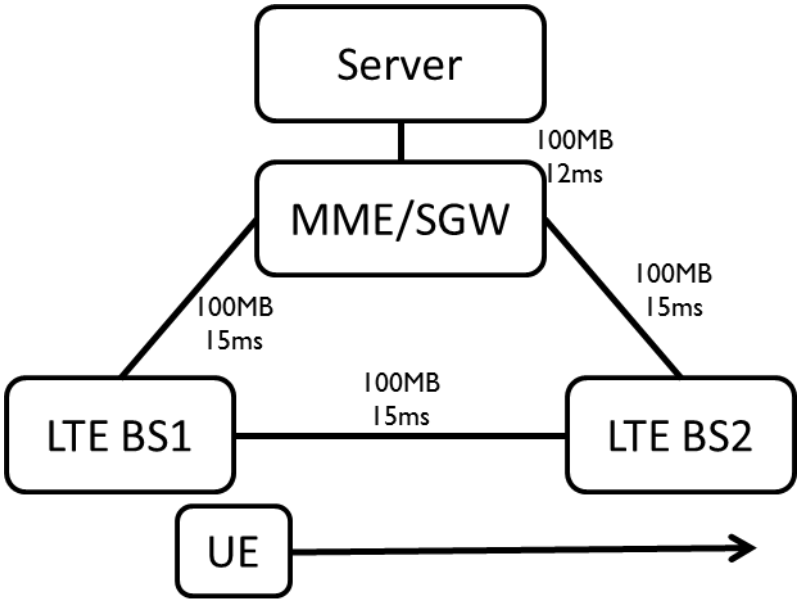


図 5.1 実験トポロジー1

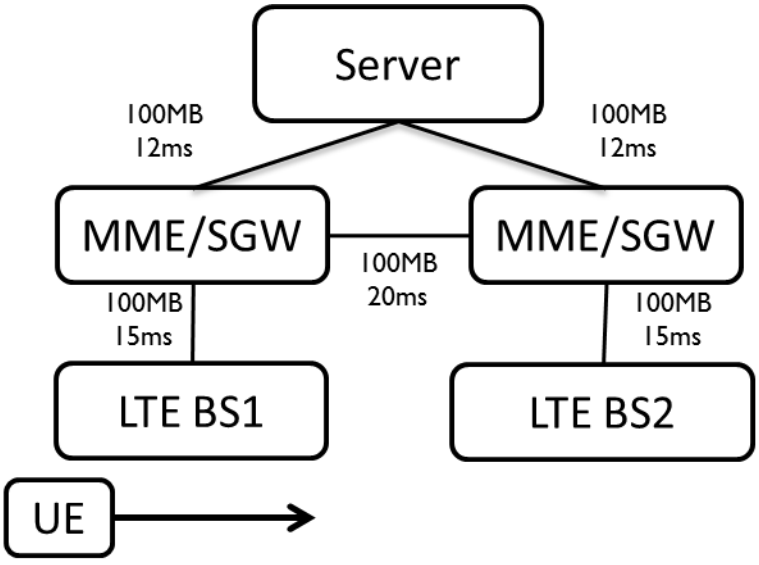


図 5.2 実験トポロジー2

シミュレーション実験開始時、UE は LTE BS1 に Associate しており、LTE BS2 方向に移動していく。移動している間、UE は Server からストリーミングデータをダウンロードし続け、ストリーミングがハンドオーバーの影響をどの程度受けるかを調査する。なお、ストリーミングデータは HTTP Live Streaming か DASH を利用してダウンロードするものとし、ライブストリーミングを想定して実験を行うため、分割ファイルの長さは 1[s]とする。また、表 5.1 は Server に用意された動画の種類を、表 5.2 はシミュレーションのパラメータを示したものである。なお、表 5.1 中の *videolevel* は 5.3.1 節で後述されるレートアダプテーションアルゴリズムで利用するパラメータである。

表 5.1 Server に用意されている動画の種類

Quality	Bitrate	videolevel
low	200kbits/s	2
middle	400kbits/s	4
high	800kbits/s	8

表 5.2 シミュレーションパラメータ

the number of Resource Block	50
modulation method	64QAM
bandwidth	10MHz
frequency	2.5GHz
queue size	100kbytes/1000packets

5.2 シミュレーション実験で利用するパラメータについて

本研究では、シミュレーション実験によってストリーミングがハンドオーバーの影響をどのように受けるかについて調査している。シミュレーション実験で利用しているパラメータのうち、有線ネットワーク部の遅延値と、ドメイン間ハンドオーバー発生時の認証遅延の値については、実際にスマートフォンとアプリケーションを利用した実験の結果を利用した。次節以降でその導出方法について述べる。

5.2.1 ネットワークの遅延の値の導出

有線部の遅延値の導出方法を以下に示す。スマートフォンとして、NTT docomo 社から販売されている、galaxy s III(OS:Android4.0.4)を利用する。

5.2.1.1　SGW・PGW・基地局の IP アドレスの推測

図 5.3 実験に利用した地域

まず、図 5.3 に示した A～H の各地点において、TraceRoute アプリを利用して、TraceRoute コマンドで当研究室サーバーまでのネットワーク経路を調査する。図 5.4 はその結果を示したものである。なお、当然 TraceRoute により具体的な IP アドレスが表示されるが、本論文では各企業に配慮して、IP アドレスの数値の掲載は控え、IP アドレス A、IP アドレス B のように表現する。

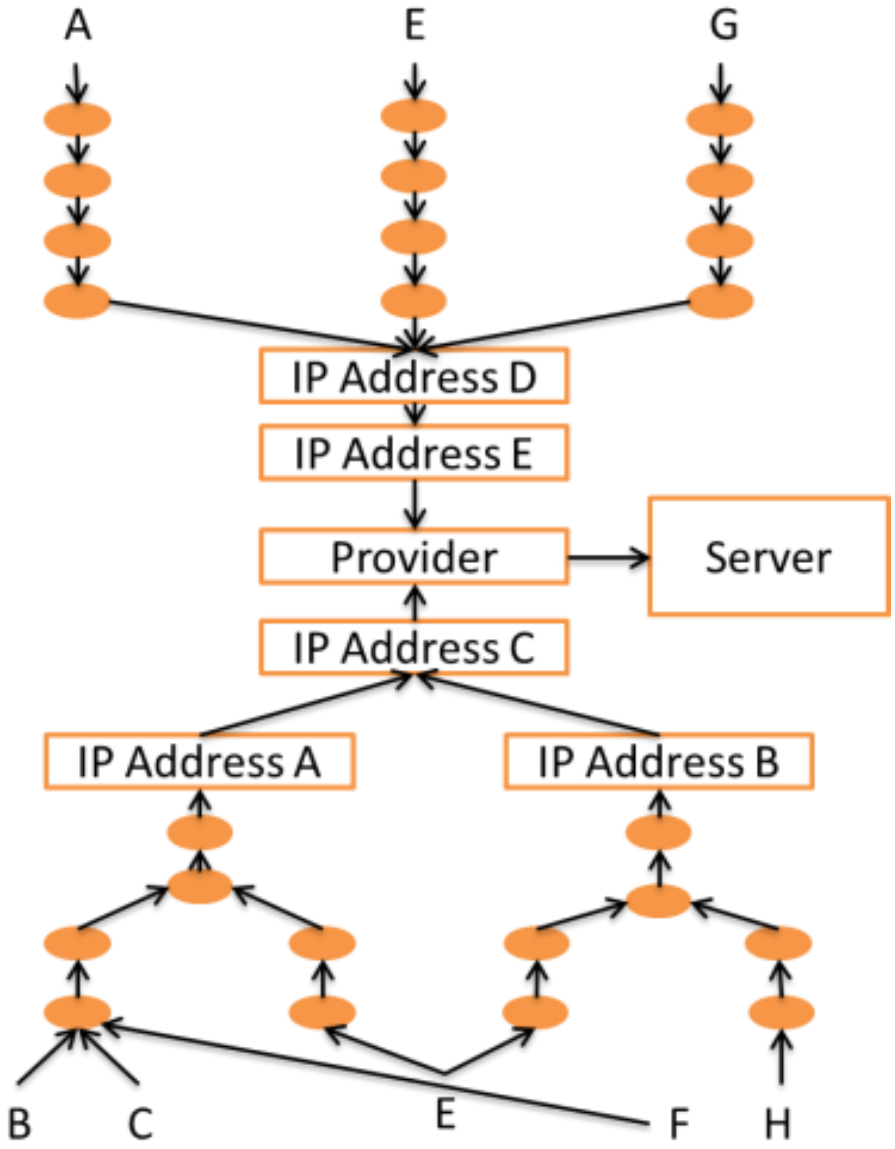


図 5.4 ネットワーク経路調査結果

ネットワーク経路を調査するのと同時に、TraceRoute の結果から確認できた IP アドレスがどの企業に属するものか[33]を利用して調査した。その結果、1hop 目から 6hop 目までが携帯キャリア会社に属するものだった。そのため、この 1hop 目から 6hop 目までが携帯キャリア会社のコアネットワークと考える。また、[29][34]よりユーザデータの転送自体にはMMEやPCRF等が関わる事がなく、このコアネットワークに含まれるのはSGW、PGW、基地局であると考えられる。

上記の点に加え、LTE の通信においてユーザデータの転送には SGW や PGW が必ず経由されるという事から、IP アドレス C、E を PGW、IP アドレス A、B、D を SGW のアドレスと判断できる。また、UE のデータを一番初めに受信するのは基地局だと考えられるので、

1hop 目は基地局の IP アドレスだと判断する。

5.2.1.2　遅延時間の導出

前節で SGW や PGW のものと予測した IP アドレスに対して、Ping! Up アプリ[35]を利用して ping コマンドで遅延時間を計測した。図 5.5 に Ping! Up アプリの実行結果例を、表 5.3 に測定結果を示す。なお、Ping の実行結果は 10[ms]程度から 4[s]と大きく揺れる。そのため、その時の環境下で安定して測定できた結果として、平均偏差が 15[ms]以内で測定できた結果を遅延時間として採用する。

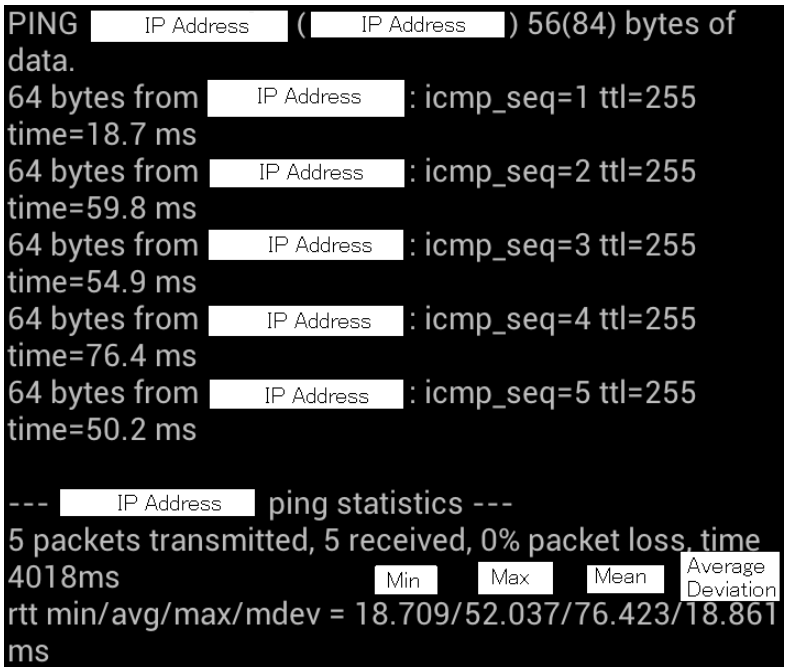


図 5.5 Ping! Up 実行結果例

表 5.3 遅延時間測定結果

	基地局	2hop	3hop	4hop	SGW	PGW	Lab Server
Route1 (fromE)	70	73	78.3	105.7	107.8	測定不能	134.7
Route2 (fromE)	84.8	87.4	86.7	86.8	94.6	102.5	134.6
Route3 (fromE)	64.5	80.3	96.8	73.4	113.5	測定不能	129.3
Route4 (fromH)	70.2	78.9	78.2	81.3	97.2	測定不能	107.9

表 5.3 から、基地局から SGW までの遅延時間は平均すると約 15[ms]、SGW から研究室のサーバーまでは平均すると 12[ms]と求める事ができる。この 2 つの値をシミュレーション実験における有線部の遅延時間として採用している。

この値は前述の通り、平均偏差が 15[ms]以内に収まっている場合の結果のみを採用したものである。そこで、この値が実際に平均値として採用できるものかどうか、確認するため、もう 1 度、同様の実験を行った。ただし、この実験では、図 5.4 でいう、E 地点から IP アドレス A、C を通るパターン(パターン 1)、E 地点から IP アドレス D と E を通るパターン(パターン 2)、H 地点から IP アドレス B、C を通るパターン(パターン 3)の 3 つのパターンについて調査する事とする。また、Ping コマンドの試行回数は 150 回である。図 5.6～図 5.8 はそれぞれ、パターン 1 における基地局までの遅延時間分布、SGW までの遅延時間分布、研究室サーバーまでの遅延時間分布を示したものである。

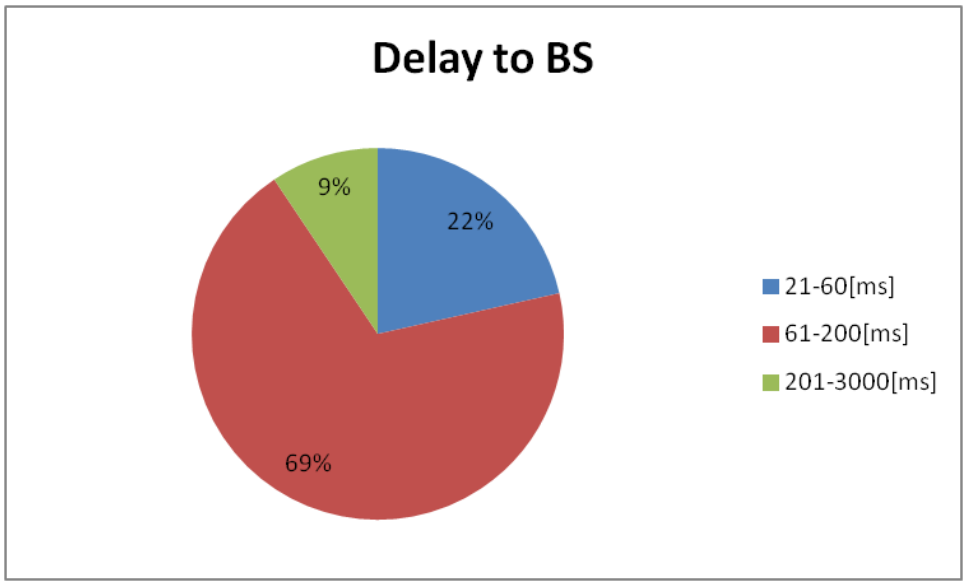


図 5.6 基地局までの遅延時間分布

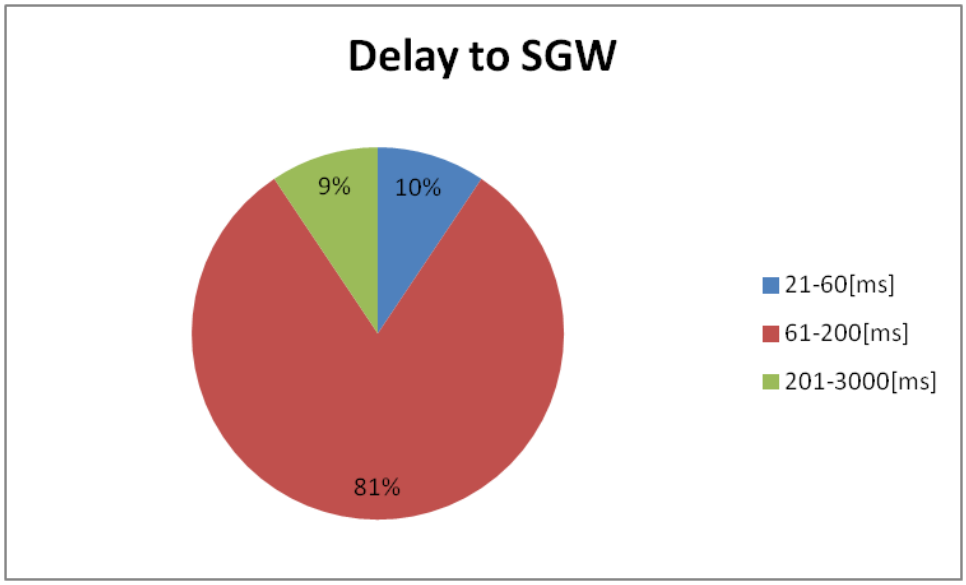


図 5.7 SGW までの遅延時間分布

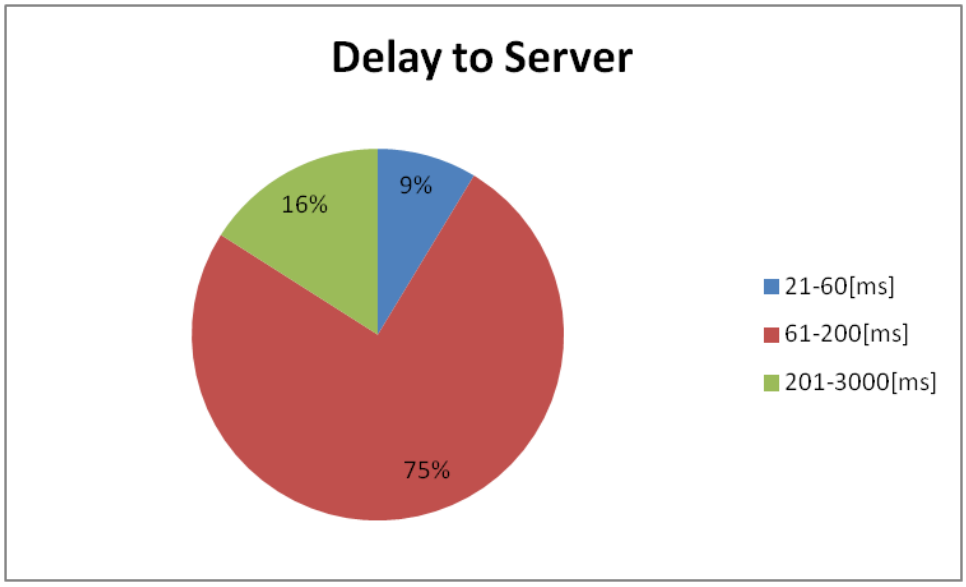


図 5.8 研究室サーバーまでの遅延時間分布

図 5.6 から図 5.8 より、UE からホップ数が少ない程遅延時間が少なくなる傾向にある事がわかる。例えば、遅延時間が 21[ms]から 60[ms]で収まる割合が、基地局までの場合は、全体の 22[%]を占め、SGW までの場合は 10[%]、サーバーまでの場合は 9[%]と徐々に減っている。逆に遅延時間が 61[ms]かかる割合は、サーバーまでが 91[%]、SGW までが 90[%]、基地局までが 78[%]となっている。パターン 2、3 の場合の結果は割愛するが、パターン 1 と似たような結果である。また、遅延時間測定結果を表 5.4 に示す。

表 5.4 遅延時間測定結果(Ping150 回による平均)

	BS	SGW	PGW	Potomac
pattern1	92	126.4	Unmeasurable	162.3
pattern2	102	124.9	126.2	146.5
pattern3	107	139	Unmeasurable	184

表 5.4 より BS—SGW 間と SGW—甲藤研サーバー間の平均遅延を求めなおすと、前者が 14.8[ms]、後者が 17.0[ms]となり、平均偏差が 15[ms]以内の値を採用した場合(それぞれ 15[ms]、12[ms])とほぼ変わらない結果となっている。このことから先ほど求めた結果はネットワークの平均の混雑具合を反映した結果として採用しても問題ないと考えられる。

5.2.2 SGW 間の遅延時間の導出

接続中の SGW までの遅延時間と、隣接していると考えられる SGW までの遅延時間を測定し、両者の差から SGW 間の遅延時間を予想している。測定結果を表 5.5 に示す。

表 5.5 SGW 間の遅延時間測定結果

Measurement Place	Connected SGW	Nearby SGW	Difference (Delay)
E	IP アドレス A	IP アドレス B	10.1
E	IP アドレス A	IP アドレス D	22.3
H	IP アドレス B	IP アドレス A	17.8
H	IP アドレス B	IP アドレス D	25.8

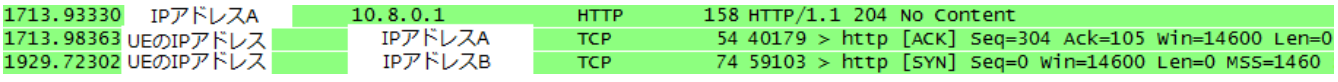
表 5.5 から SGW 間の平均遅延時間は約 20[ms]と求めることができる。

5.2.3　ドメイン間ハンドオーバー発生時の認証遅延の導出

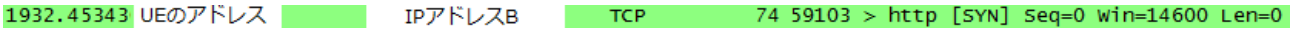
本研究では、ドメイン内ハンドオーバーだけでなく、異なる SGW に接続しなおす必要のあるドメイン間ハンドオーバーに関してもシミュレーション実験を行う。ドメイン間ハンドオーバーでは UE が再接続する際に、MME が UE を再認証するステップが加わるため、認証遅延がドメイン内ハンドオーバーと比べると、余計にかかることになる。その認証遅延が実際に LTE ネットワークではどのくらいかかるのか、以下の方法で予測・導出している。

5.2.3.1　MME の IP アドレス推測

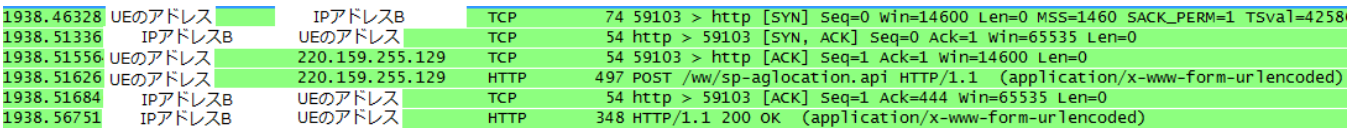
図 5.3 中の H 地点から E 地点まで移動し、その間 tPacketCapture アプリ[36]を利用して、パケットキャプチャを行う。[36]によって生成された pcap ファイルの情報から認証遅延値を導出する。図 5.9 はパケットキャプチャの結果を示したものである。



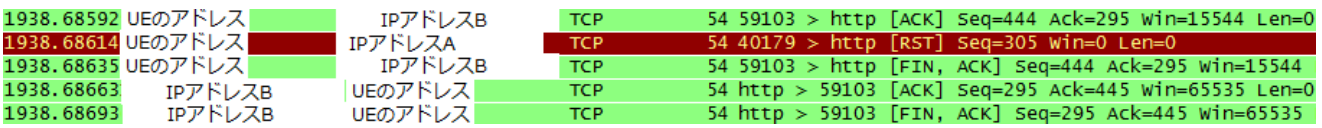
(a)



(b)



(c)



(d)

図 5.9 パケットキャプチャ結果

図 5.9(a)はハンドオーバー発生直前と、直後のパケットキャプチャの結果を示したものである。[29]より HO 直後の接続手順は UE が MME にアタッチ要求を行い、端末の認証、位置登録の流れでスタート判断できる。そのため、UE がハンドオーバー直後、最初に接続を

試みる IP アドレスであり、[33]で調査した結果、携帯キャリア会社に属する IP アドレスである IP アドレス B が MME の IP アドレスと予測できる。この IP アドレス B を利用して次節以降で認証遅延を求める。

5.2.3.2　認証遅延の導出

UE は図 5.9(a)の 3 行目、(b)、(c)の 1 行目と、3 回 MME(IP アドレス B)とのセッション確立を試みているが、実際に成功したのは 3 回目のみである。そこで、今回はこの 3 回目に UE が SYN パケットを送信した 1938.46328[s]を MME との接続を開始した時間とする。この後、UE が MME からの SYN/ACK を受信(1938.51336[s])、UE が ACK を送信(1938.51556[s])して、3 ハンドシェイクが成功する。ここまでの時間が約 52[ms]である。
次に図 5.9(c)の 4 行目で UE が POST プロトコルで sp-aglocation.api を利用し、位置登録を行うステップがある(1938.51626[s])。この通信に対して、MME が ACK を返し(1938.51684[s])、HTTP セッションが確立される(1938.56751[s])。ここまでの手順で位置登録が完了すると考えると、位置登録が完了するまでの時間は約 51[ms]である。
つまり、ドメイン間ハンドオーバーで MME と UE のセッションが切断され、再認証するまでの遅延時間は 103[ms]と予測できる。

5.3　実験で利用する Dynamic Adaptive Streaming over HTTP について

前述の通り、本研究ではシミュレーション実験により、LTE ハンドオーバーが発生する際の HTTP Live Streaming と DASH の性能評価を行っている。3.6.4 節で述べたように、これまでに数多くの DASH や Dynamic Adaptive Streaming に関しての研究がされてきたが、本研究では[37]の帯域推定・アダプテーション方法を利用してシミュレーション実験を行う。

5.3.1　帯域推定・レートアダプテーション方法[37]

前述の通り、本研究では[37]を参考にした帯域推定・アダプテーション方法をシミュレータである Scenargie に実装してシミュレーション実験を行っている。[37]のレートアダプテーションアルゴリズムは、クライアントが分割されたファイルをダウンロードするのにかかった時間(*downloadtime*)が基準となっている。クライアントは分割ファイルのダウンロードにかかった時間(*downloadtime*)と分割ファイルの再生時間(*MSD*)を式(5.1)のように比較する。

$$\mathrm{check} = \frac{\mathrm{MSD}}{\mathrm{downloadtime}} \tag{5.1}$$

この *check* の値に基づいて式(5.2)のようにアダプテーションを行う。なお、*videolevel* はストリーミングデータの品質を示したもので、本研究では主にストリーミングデータのビットレートの値が格納される。

videolevel

$$= \begin{cases} videolevel * check, if\ check < switchdownfactor \\ nextvideolevel, else\ if\ check > 1 + switchupfactor \\ nowvideolevel, else \end{cases} \quad (5.2)$$

switchdownfactor は *videolevel* を下げるかどうかを決定する閾値である。*downloadtime* の値が *MSD* の値を超えてしまう事をどこまで許容するかを決定する値であり、この値が大きい程、その許容範囲は小さくなる。*switchupfactor* は *videolevel* を上げるかどうかを決定する値で式(5.3)のように定義される。

$$\text{switchupfactor} = \frac{\text{nextvideolevel} - \text{nowvideolevel}}{\text{nowvideolevel}} \quad (5.3)$$

nowvideolevel は現在選択しているストリーミングデータの *videolevel* を示しており、*nextvideolevel*は現在選択している *videolevel*よりも一段階上の *videolevel*を示している。つまり、例えば現在選択している *videolevel* が 2(低画質)であれば、次の *videolevel* は 4(中画質)となる。式(5.2)からわかるように、ネットワークの状況が悪化した場合には、動画の再生が停止するといったような状況を避けるためにすぐに *videolevel* を低下させ、反対にネットワークの状況が良くなった場合には、*videolevel* を上げ、ユーザーが、より品質の良い動画を再生する事を可能にする。ネットワークの環境が良くなった後、急にネットワーク環境が再び悪化した場合等で、動画の再生が停止するといった状況が考えられる。本研究では、動画の再生が停止しない事を第一に考えているため、*videolevel* を上げる場合は急激でなく、あえて緩やかにしている。

5.4 実験結果

6 つのシナリオを用意し、シミュレーション実験を行っている。まず、最初の 2 つのシナリオは、1 つの UE が、基地局 LTE BS1 から基地局 LTE BS2 に移動し、かつ、基地局 LTE BS2 に複数の UE がもとから存在するパターンで、ドメイン内ハンドオーバーを想定したものをシナリオ 1、ドメイン間ハンドオーバーを想定したものをシナリオ 2 とする。シナリオ 1、2 は移動先の基地局が混雑している場合を想定しており、LTE BS2 にいる UE の数を 0、30、50 と変化させる。実験トポロジーとしては図 5.10、図 5.11 のような形になる。あとの 2 つは、複数の UE が LTE BS1 から LTE BS2 に同時にハンドオーバーするパターンで、ドメイン内ハンドオーバーを想定したものをシナリオ 3、ドメイン間ハンドオーバーを想定したものをシナリオ 4 とする。シナリオ 3 や 4 は、例えばバスや電車に乗り、ストリーミングを行っているユーザーが同時にハンドオーバーする場面等を想定している。なお、同時にハンドオーバーする UE の数を 1、30、50 と変化させる。実験トポロジーとしては図 5.12、図 5.13 のような形になる。残り 2 つはこれまでのシナリオを併せたもので、シナリオ 1、2 の状態に加えて移動元の基地局にも複数の UE がいるパターンでそれぞれシナリオ 5、シナリオ 6 とする。実験トポロジーとしてはそれぞれ図 5.14、図 5.15 のような

形である。なお、*MSD* と *switchdownfactor* の値はライブストリーミングを想定して 1[s]、0.83 に設定する。また、ドメイン間ハンドオーバー時の認証にかかる時間を 5.2.3.2 節で予測した通り、103[ms]とする。

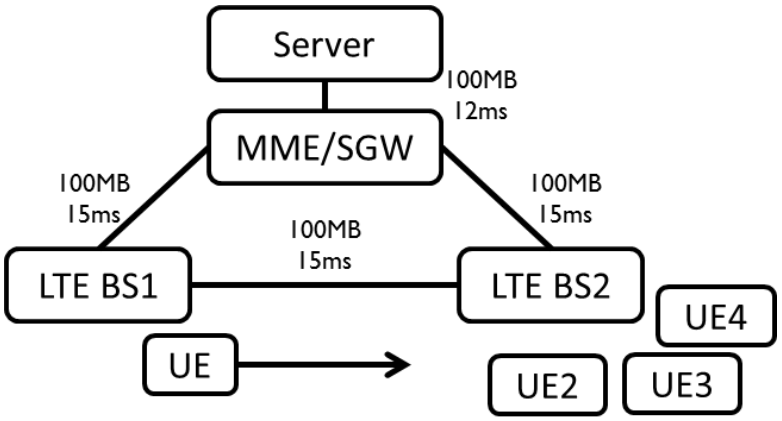


図 5.10 実験トポロジー例(シナリオ 1)

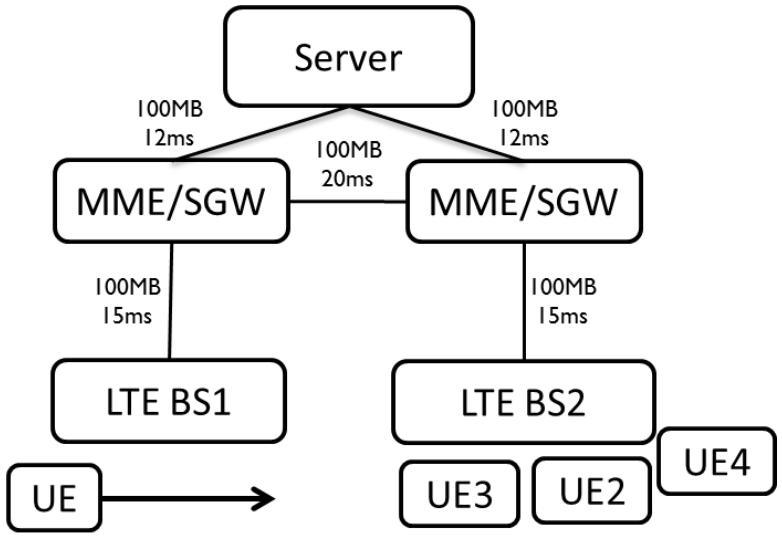


図 5.11 実験トポロジー例(シナリオ 2)

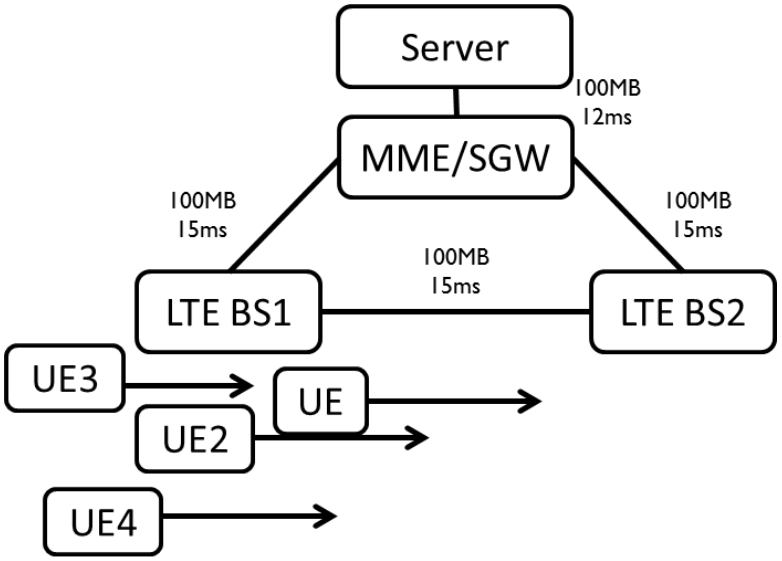


図 5.12 実験トポロジー例(シナリオ 3)

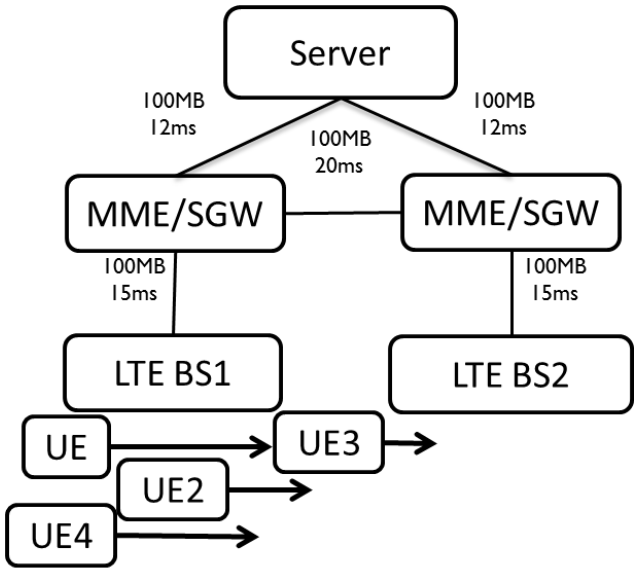


図 5.13 実験トポロジー例(シナリオ 4)

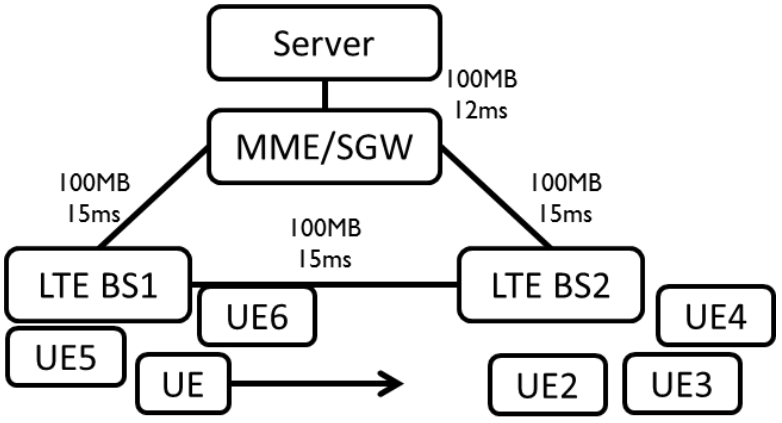


図 5.14 実験トポロジー例(シナリオ 5)

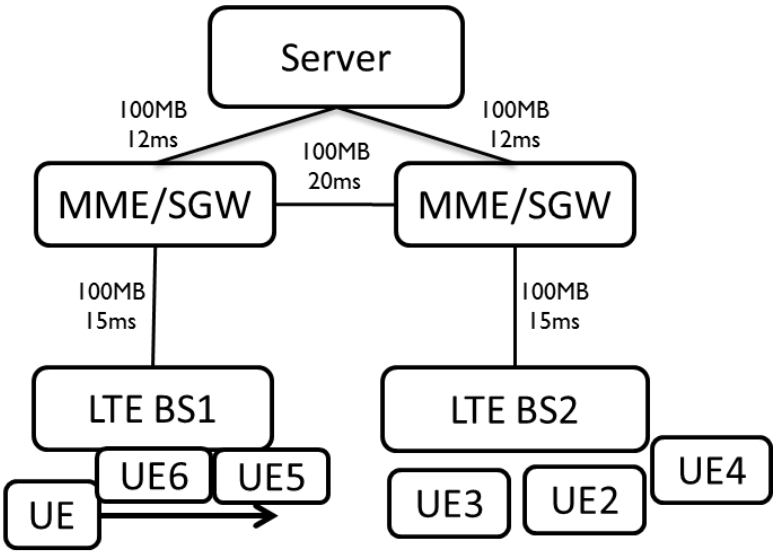


図 5.15 実験トポロジー(シナリオ 6)

5.4.1 移動先の基地局が混雑している場合(シナリオ 1、2)の実験結果

図 5.16 はシナリオ 1 における分割ファイルのダウンロード時間を、図 5.17 はシナリオ 2 における分割ファイルのダウンロード時間を比較したものである。なお、X 軸は BS2 に存在する UE の数を示しており、Y 軸はダウンロード時間を示している。また、図中の低画質、中画質、高画質は HTTP Live Streaming でストリーミングデータをダウンロードした場合に相当し、Adaptive は DASH でダウンロードした場合を示す。低画質、中画質、高画質が対応するビットレート等については、すでに、5.1 節で表 5.1 に示した通りである。

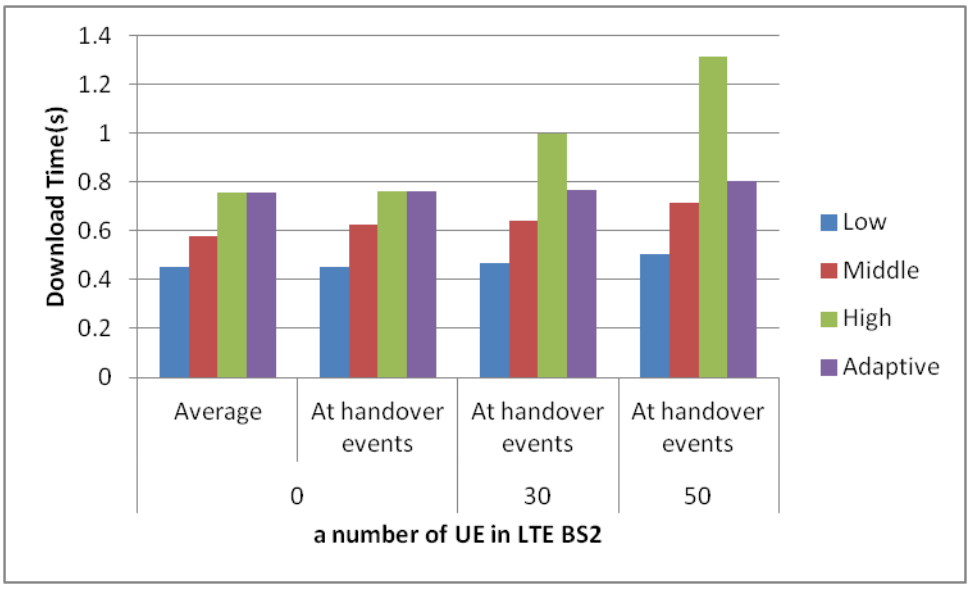


図 5.16 シナリオ 1 におけるダウンロード時間比較

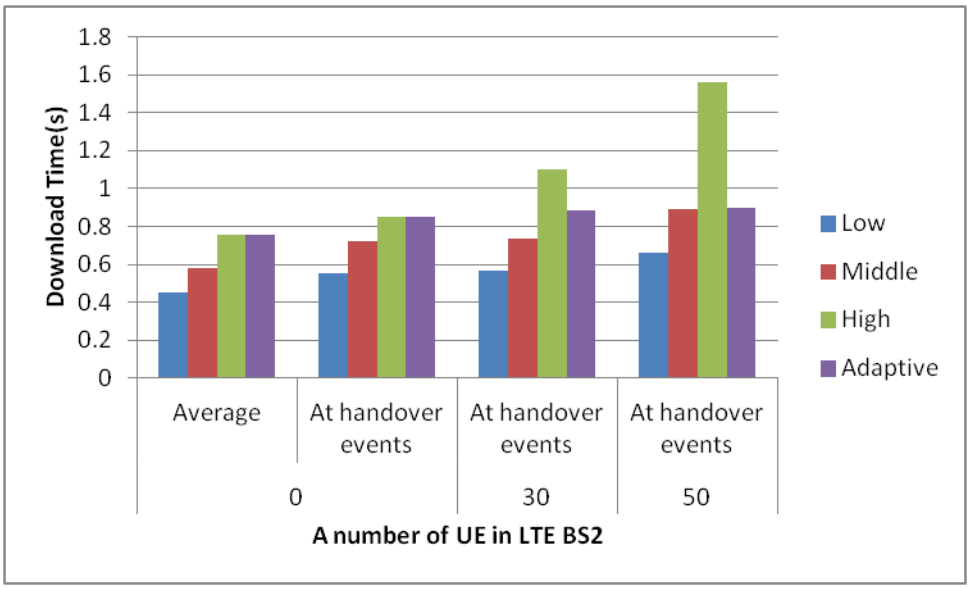


図 5.17 シナリオ 2 におけるダウンロード時間比較

図 5.16、図 5.17 から、HO が発生しない場合(平均)に比べて、HO 発生時には、分割ファイルのダウンロード時間が増加する事がわかる。これはハンドオーバー時のパケット転送の手順に関係がある。通常、ハンドオーバーが発生時には、移動先の基地局と接続しなおすまで、UE は移動元の基地局とパケット通信を行う事ができず、移動元の基地局のキューにパケットが格納されることになる。そして、格納されたパケットは LTE が採用する X2 ハンドオーバーの場合、移動元の基地局から移動先の基地局に直接され、移動先の基地局経由で UE にパケットが転送される。そのためこの実験の場合、LTE BS1 から LTE BS2 にパケットを転送する時間が増えるため、ダウンロード時間が増加している。また、移動先の基地局にいる UE 数が増えるほど、ダウンロード時間も増加している。これは UE 数が増えるほど、OFDMA におけるサブキャリアの割り当て数等が減少し、帯域速度が減少する事と関係がある。また、LTE BS1 から LTE BS2 では転送パケットが優先的に送信されるが、LTE BS2 から UE まではパケットが最優先で送信されるわけではなく、UE 数が多く混雑しているときほど、パケットが UE に到着する時間が多くなる事も原因の 1 つである。また、図 5.16 と図 5.17 を比較すると、図 5.17 の方がダウンロード時間が大きいことがわかる。ドメイン間ハンドオーバーではドメイン内ハンドオーバーと異なり、基地局同士でパケットが転送する事ができず、必ず、コアネットワークを経由しなければならない事と認証遅延が余計にかかる事が原因である。

移動先の基地局に複数の UE がいるシナリオではダウンロード時間は、上記のような原因で増える。しかし、シナリオ 1、2 において複数の UE が 50 人いる場合に、HTTP Live Streaming で高画質の動画をダウンロードする場合と、シナリオ 2 において複数の UE が 30 人いる場合に HTTP Live Streaming で高画質の動画をダウンロードする場合を除いて、ダウンロード時間は 1[s]以内に収まっている。そのため、ほとんどのユーザーは動画が停止

する等の障害が発生する事なく、動画を再生することができている。

図 5.18 はシナリオ 1 において、LTE BS2 にいる UE の数が 0、30、50 人の時の HO 時のパケット 1 つ 1 つの片方向遅延分布と HO が発生しない時の遅延分布を比較したものである。また、図 5.19 は図 5.18 から HO 時に転送されたパケットについてのみ抽出した片方向遅延分布である。

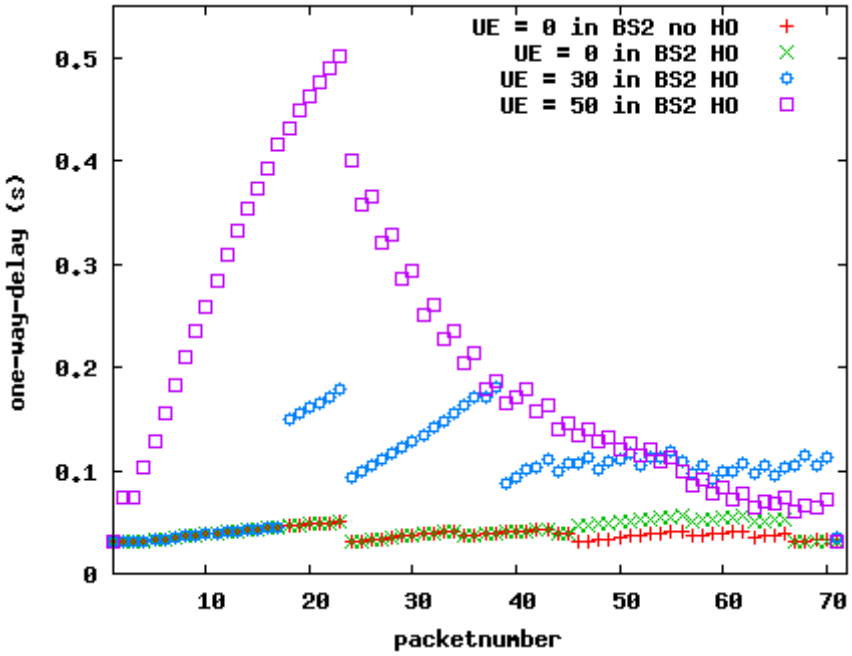


図 5.18 片方向遅延分布

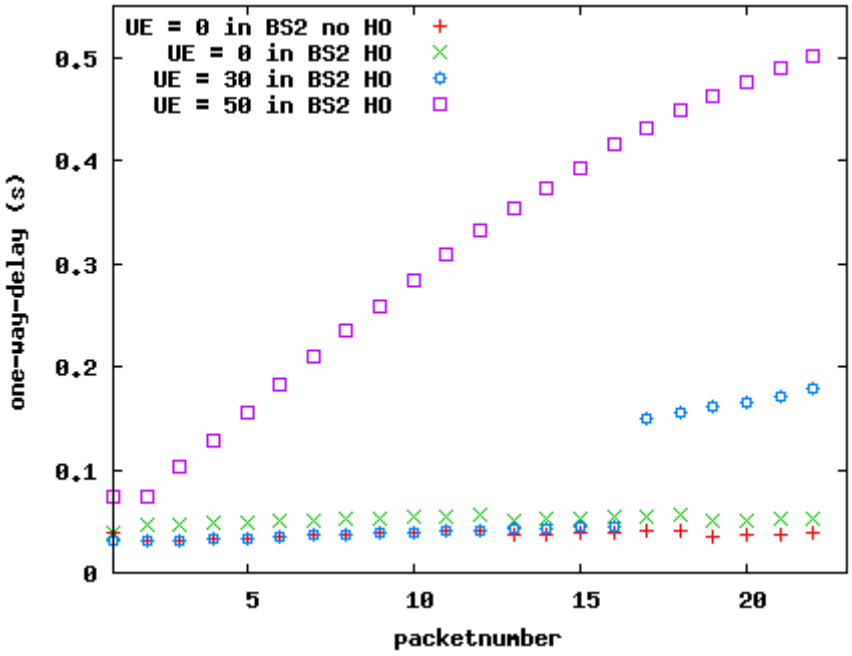


図 5.19 HO 時に転送されたパケットの片方向遅延分布

図 5.18 と図 5.19 から分かるように、移動先の基地局にいる人数が多い程、パケットの片方

向遅延が大きいことが分かる。図 5.20 と図 5.21 はシナリオ 1、2 におけるスループットを比較した結果である。なお、X 軸は移動先の基地局にいる UE 数、Y 軸はスループット(kbps)を示している。

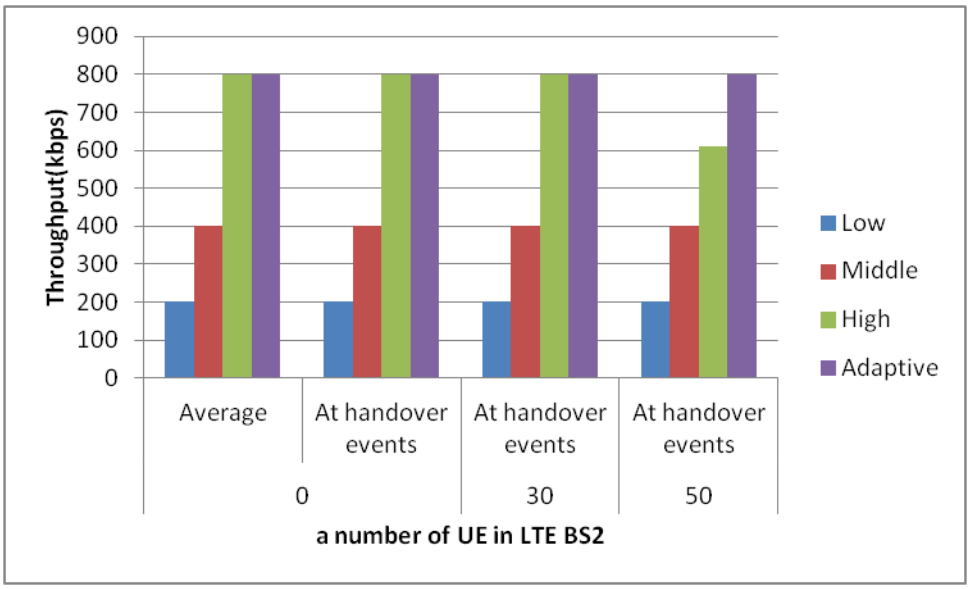


図 5.20 シナリオ 1 におけるスループット比較

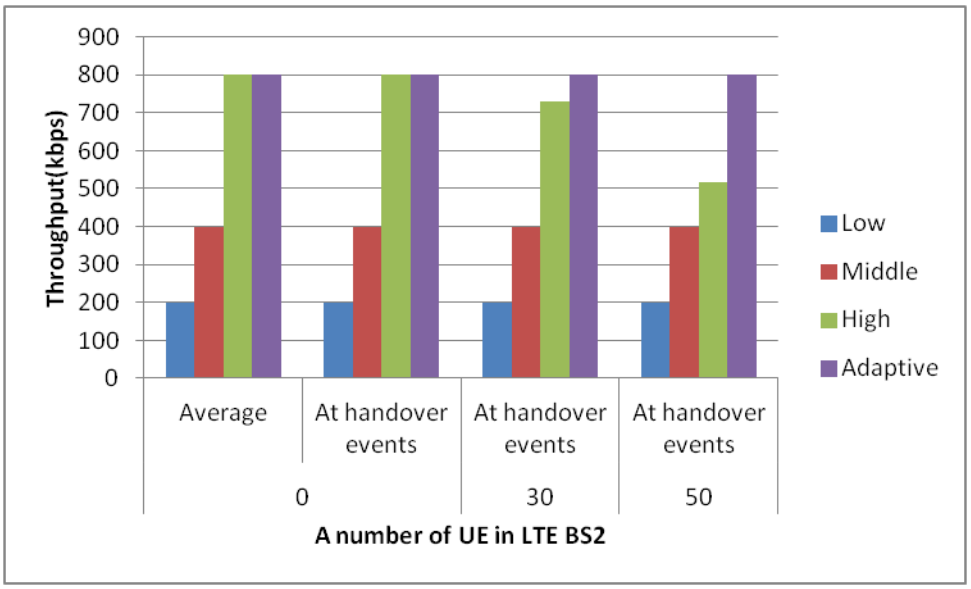


図 5.21 シナリオ 2 におけるスループット比較

前述の通り、ほとんどの場合において UE は 1[s]以内に分割ファイルのダウンロードを完了する事ができる。そのため、スループットの値は各画質のビットレートに等しい場合が多い。スループットが各画質のビットレートに等しくない場合はダウンロードが 1[s]以内に完了しなかった場合を示す。また、ドメイン間ハンドオーバーの場合、認証遅延等が加わるため、ドメイン内ハンドオーバーの場合に比べてスループットが若干低下する事がわかる。

5.4.2　同時に複数の UE がハンドオーバーする場合(シナリオ 3、4)の実験結果

図 5.22、図 5.23 はシナリオ 3、4 におけるダウンロード時間を比較したものを示したものである。

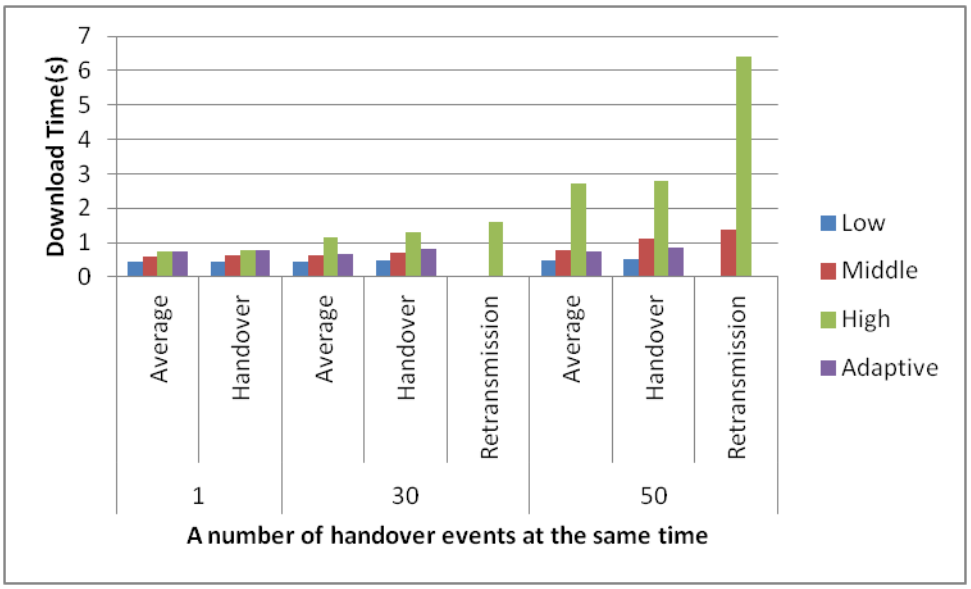


図 5.22 シナリオ 3 におけるダウンロード時間比較

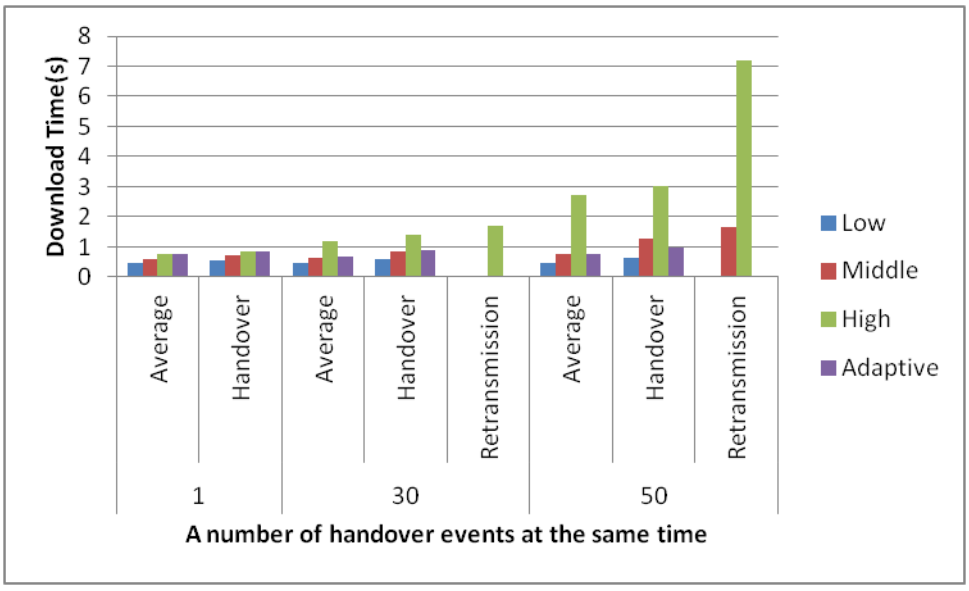


図 5.23 シナリオ 4 におけるダウンロード時間比較

図 5.22、図 5.23 から同時にハンドオーバーする UE の数が増えるほど、ダウンロードの時間が増加する事がわかる。また、HTTP Live Streaming で高画質の動画をダウンロードする場合、UE の数が 30 以上なら、ダウンロード時間が 1[s]を超えており、ユーザーが動画の停止といった障害に遭遇してしまう事がわかる。また、HO 時に LTE BS1 のキューに急激にパケットがたまる事によって LTE BS1 でのパケット損失が発生してしまうため、UE

の中にはパケット損失による再送が発生し、深刻な動画の停止という状況に遭遇してしまう。例えば、同時に 50 の UE がハンドオーバーする場合は、7[s]以上も動画が停止してしまう事がわかる。このように HTTP Live Streaming の場合、動画のスムーズな再生が困難な状況になってしまうが、MPEG-DASH の場合、ネットワークの状況に合わせて、動画のビットレートを動的に変化させるため、再送や動画のダウンロードが間に合わないといった事態を防ぐことができる。表 5.6 はシナリオ 3 における、再送に遭遇するユーザーの割合を示したもの、表 5.7 はシナリオ 4 における再送に遭遇するユーザーの割合を示したものである。MPEG-DASH を利用した場合、HTTP Live Streaming を利用した場合に比べ再送にまきこまれるユーザーの割合を大きく減少させることができている事がわかる。

表 5.6 再送に遭遇するユーザーの割合

	HTTP Live Streaming	MPEG-DASH
Middle	38%	0%
High	82%	0%

表 5.7 再送に遭遇するユーザーの割合

	HTTP Live Streaming	MPEG-DASH
Middle	40%	0%
High	84%	0%

また、図 5.24 と図 5.25 はシナリオ 3、4 におけるスループットを比較したものである。

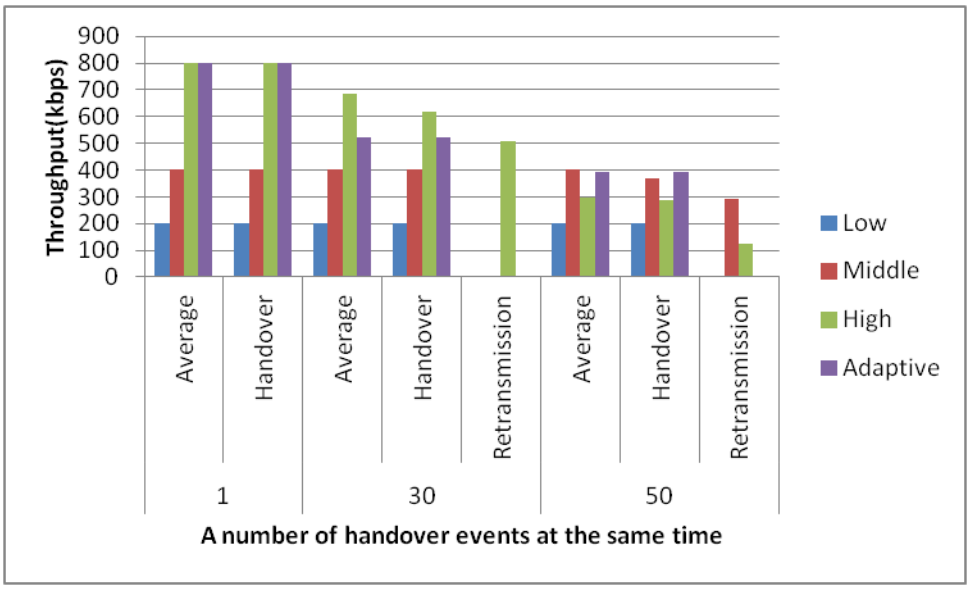


図 5.24 シナリオ 3 におけるスループット比較

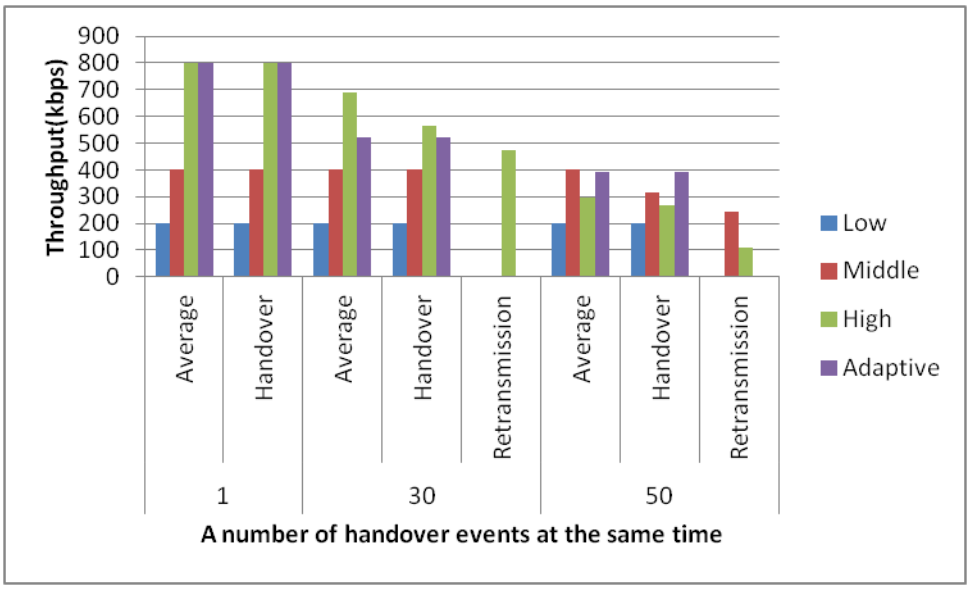


図 5.25 シナリオ 4 におけるスループット比較

図 5.24 と図 5.25 から同時にハンドオーバーする UE の数が増加するほど、スループットが低下している事がわかる。特に、高画質の動画を HTTP Live Streaming でダウンロードする場合のスループットの低下が激しい。しかし、ネットワークの状況に合わせて動画のビットレートを変化させる MPEG-DASH の場合ならばネットワークのキャパシティを超えて、動画をダウンロードしようとするユーザーが減るために、スループットの低下を抑える事ができる。

5.4.3 移動先の基地局・移動先の基地局が共に混雑しており、1 つの UE がハンドオーバーする場合(シナリオ 5、6)の実験結果

シナリオ 1 と 2 では、LTE BS1 から 1 人の UE が LTE BS2 にハンドオーバーし、かつ LTE BS2 の UE 数を 30 人、50 人に変化させて実験を行っていた。今回はさらに移動元の基地局である LTE BS1 に UE をさらに 30 人加え、LTE BS2 にいる UE 数を 50 人に固定して実験を行った。前述の通り、シナリオ 5 をドメイン内ハンドオーバーの場合、シナリオ 6 をドメイン間ハンドオーバーの場合の実験結果とする。図 5.26、図 5.27 はシナリオ 5、6 におけるダウンロード時間を比較したものである。

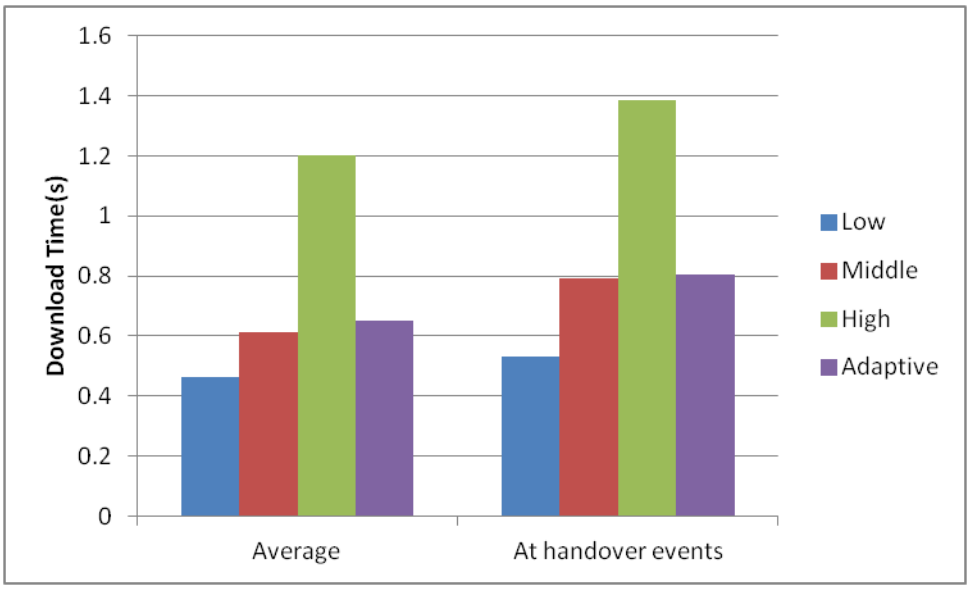


図 5.26 シナリオ 5 におけるダウンロード時間比較

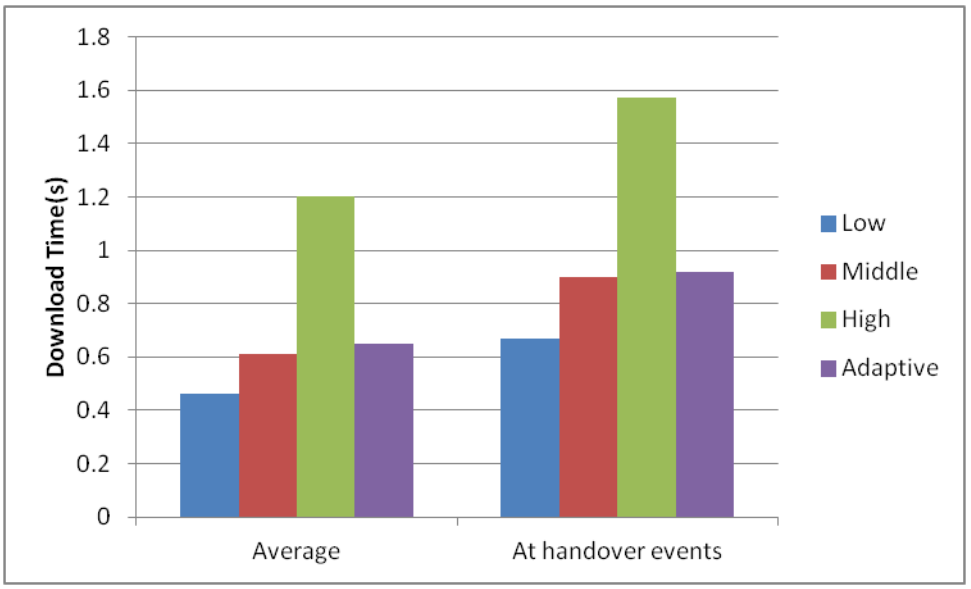


図 5.27 シナリオ 6 におけるダウンロード時間比較

シナリオ 1、2 との違いは移動元の基地局にも複数の UE がいることである。今回は移動元の基地局に 30 人の UE がいるので、平均的なダウンロード時間は図 5.22、図 5.24 における同時に 30 人の UE がハンドオーバーする場合のダウンロード時間とあまり変わっていない事がわかる。また HO 時のダウンロード時間も図 5.16、図 5.17 における LTE BS2 に 50 人の UE がいる場合とあまり変わっていない事がわかる。そのため、移動元の基地局が混雑していても、HO 時のダウンロード時間自体はそこまで変化しておらず、移動元の混雑状況自体は HO 時のダウンロード時間に大きな影響を与えていない事がわかる。図 5.28 と図 5.29 はシナリオ 5、6 におけるスループット比較を示したものである。

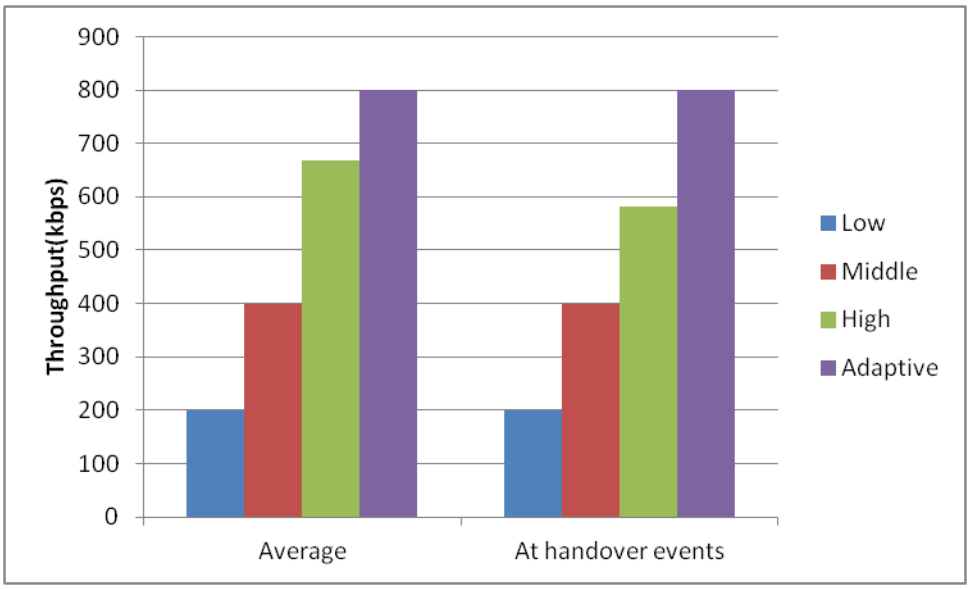


図 5.28 シナリオ 5 におけるスループット比較

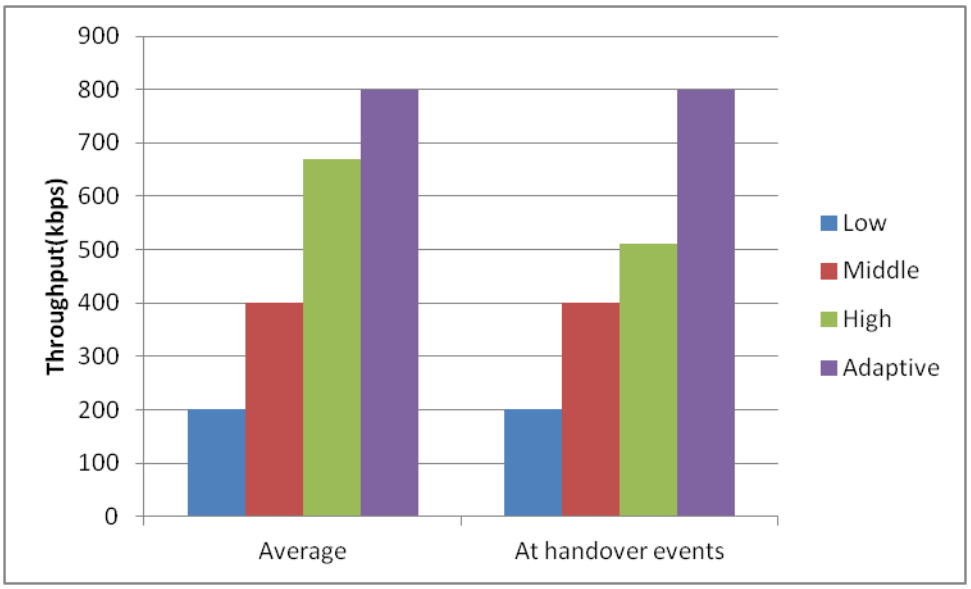


図 5.29 シナリオ 6 におけるスループット比較

5.4.1 節で述べたように、移動先の基地局が混雑しているために、OFDMA におけるサブキャリアの割り当て数が減る等の原因で、HO 時はダウンロード時間が増加する。それに伴い、HTTP Live Streaming では、高画質をダウンロードしている場合、動画の再生が間に合わない。そのため、スループットが低下する。

5.4.4 移動先の基地局・移動先の基地局が共に混雑しており、複数の UE が同時にハンドオーバーする場合(シナリオ 7、8)の実験結果

シナリオ 3、4 の場合は、LTE BS2 に UE は存在しない状況で、移動先の基地局は混雑していない状況であった。そのため、今回のシナリオ 7、8 では LTE BS2 に 30 の UE を存

在させ、LTE BS1 から同時にハンドオーバーする UE 数を 50 として実験を行っている。図 5.32 と図 5.33 はそれぞれシナリオ 7、8 におけるダウンロード時間を比較したものである。

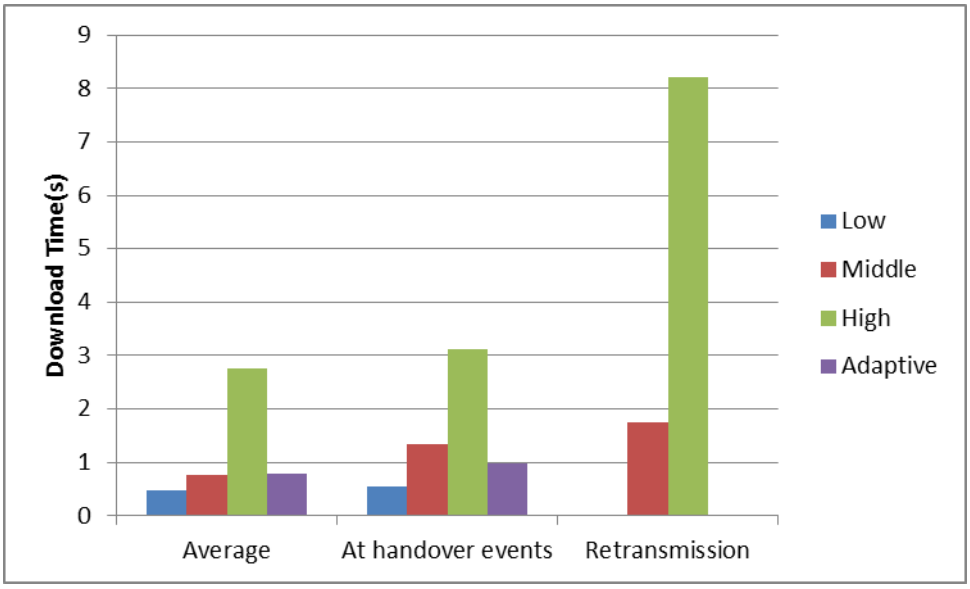


図 5.32 シナリオ 7 におけるダウンロード時間比較

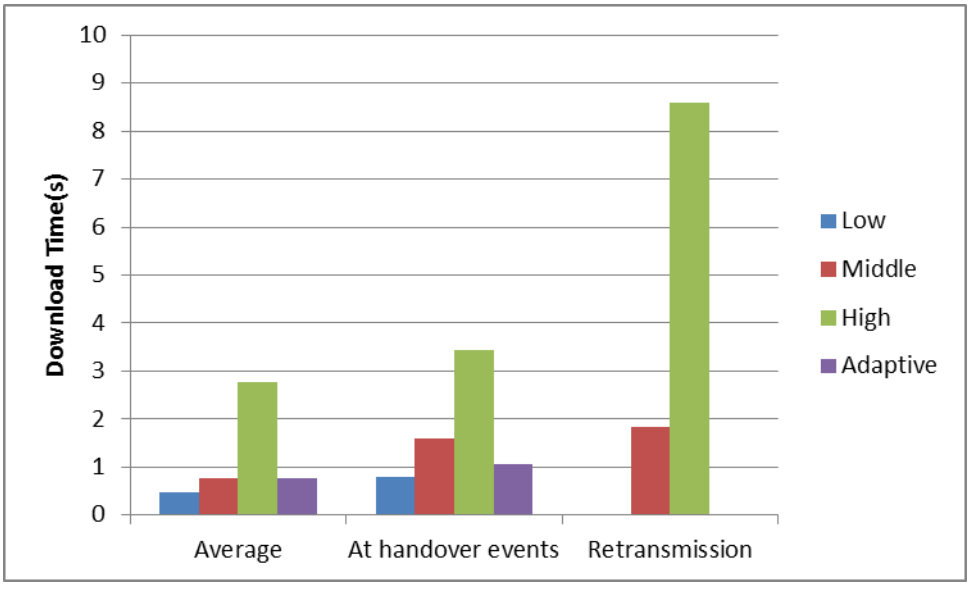


図 5.33 シナリオ 8 におけるダウンロード時間比較

シナリオ 3、4 に比べて、ダウンロード時間がさらに増加していることがわかる。シナリオ 3、4 の場合と異なり、LTE BS2 にも UE がいることにより、OFDMA におけるサブキャリア割り当て数の減少等で、通信速度が低下する。そのため、ダウンロード時間も増加してしまう。また、表 5.8、表 5.9 は再送に遭遇するユーザーの割合を示したものであるが、シナリオ 3、4 の時と同様に、ネットワークの状況に合わせてビットレート等を動的に変化さ

せる MPEG-DASH の場合、大きく再送に遭遇するユーザーの割合を減少させることができることが分かる。

表 5.8 再送に遭遇するユーザーの割合(シナリオ 7)

	HTTP Live Streaming	MPEG-DASH
Middle	54%	0%
High	80%	0%

表 5.9 再送に遭遇するユーザーの割合(シナリオ 8)

	HTTP Live Streaming	MPEG-DASH
Middle	54%	0%
High	86%	0%

図 5.34 と図 5.35 はシナリオ 7、8 におけるスループット比較を示したものである。

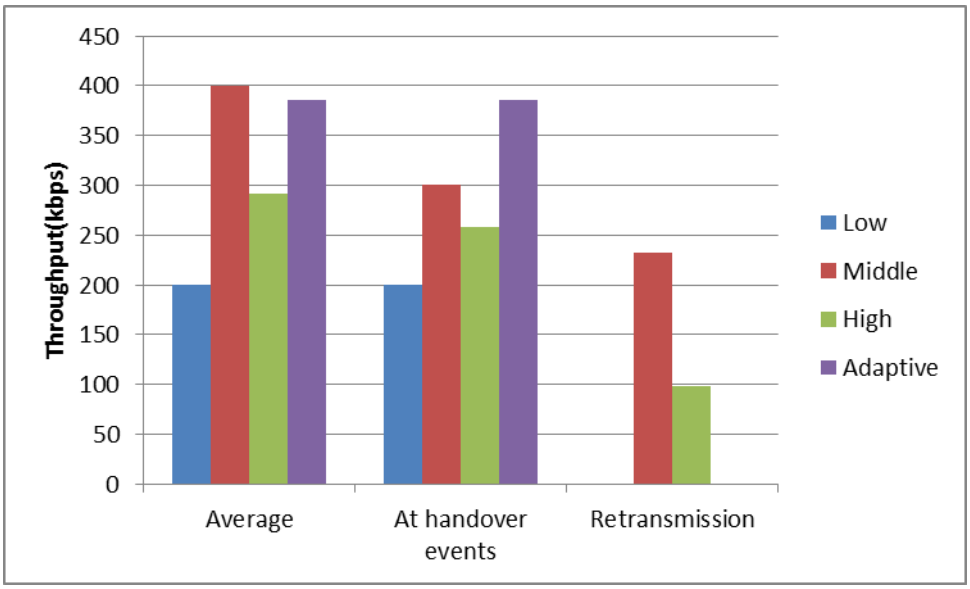


図 5.34 シナリオ 7 におけるスループット比較

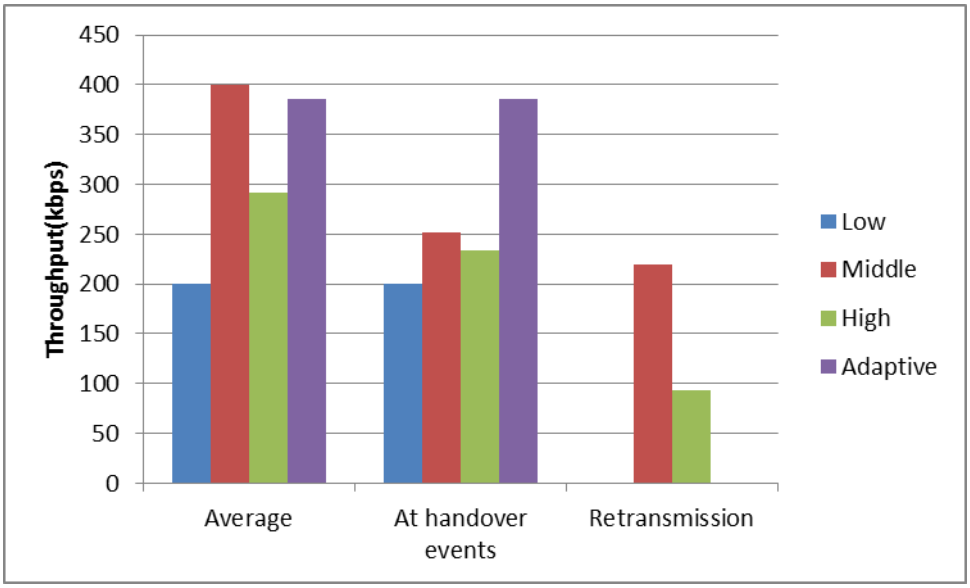


図 5.35 シナリオ 8 におけるスループット比較

図 5.34、図 5.35 から、MPEG-DASH の場合、平均的なスループットが中画質に劣っている事がわかる。ただスループットで劣っているものの、その分、MPEG-DASH の方は再送といった事態は発生していない。今回の MPEG-DASH の実装では、動画の品質を低下させる結果になってしまったとしても、各ユーザーが 1[s]以内に動画のダウンロードを完了できる事を目標としているので、スループットが今回のように劣る場合があるものの、再送という最悪な事態を防ぐことができている。ただ、シナリオ 3、4 の場合と異なり図 5.36 に示すように、シナリオ 8 の場合ハンドオーバー時に、MPEG-DASH でストリーミングを行っていても、ダウンロードが間に合わないユーザーが出てくる。

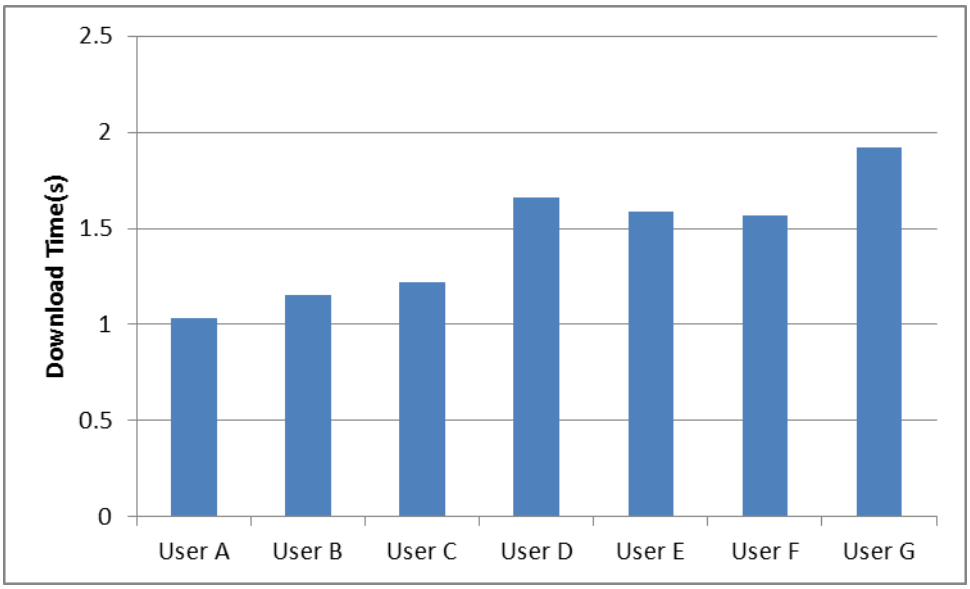


図 5.36 ダウンロードが間に合わないユーザーのダウンロード時間分布(シナリオ 8)

今回の実装では、ネットワークの環境に応じて、動画のビットレートを変化させる事でダウンロード時間が 1[s]を超える事を防いでいた。しかし、今回、LTE BS2 に 30 の UE がいるため、HO 直後から合計 80 の UE で動画をダウンロードする事になる。すると、同時に HO する UE、もともと LTE BS2 にいた UE のどちらも、80 人の UE が同一基地局にいる状態に対応する事が即時にはできず、結果としてネットワークのキャパシティを超えた状態で各 UE が動画をダウンロードしようとする。そのために、上記のように動画のダウンロードが間に合わないユーザーが出現する事態となる。

5.4.5 MPEG-DASH 利用時のユーザー間の公平性

MPEG-DASH を利用時にユーザー間で配信する動画の質はどのように変化しているかについて述べる。図 5.37 は同一基地局に 30 人のユーザーが居る場合の配信動画のレベル分布、なお、図 5.37 は各ユーザーのダウンロードする動画のレベルが安定した状態時の結果を示したものである。

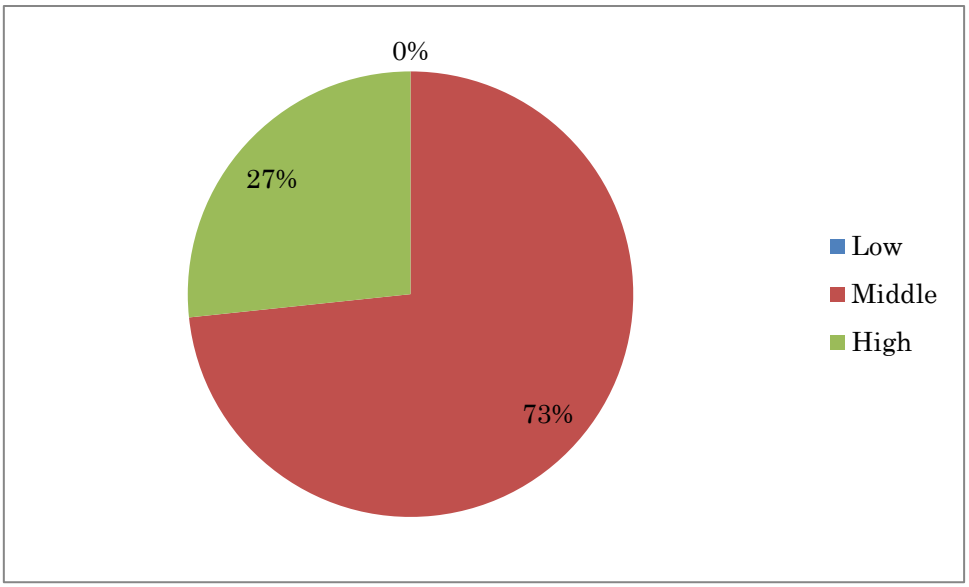


図 5.37 配信動画レベル分布(ユーザーが同一基地局に 30 人いる場合)

図 5.37 から、30 人のユーザーが同一基地局内で、MPEG-DASH を用いて動画をダウンロードすると、中画質の動画をダウンロードするユーザーと高画質の動画をダウンロードするユーザーどちらも存在する事になり、同じ環境でありながら多少の不公平が生じている事がわかる。ただ、本研究はユーザーが動画の再生を途切れることなくスムーズに行える事に重点をおいているため、このような不公平が生じてしまう結果になっても、Adaptive なストリーミングを行うべきだと主張している。

第 6 章　まとめ

6.1　総括

本論文では、LTE ハンドオーバーを想定して、シミュレータである Scenargie を利用して HTTP Live Streaming と MPEG-DASH の性能評価実験を行った。HO 時には、移動元の基地局から移動先の基地局へパケットを転送する必要があり、そのパケットを転送する時間分、ダウンロード時間は増加する。この傾向は移動先の基地局が混雑している程、顕著で、ストリーミング再生に大きな影響を与える。また、同時に複数の UE がハンドオーバーする環境下では、移動元の基地局にキューからパケットがあふれ、損失する事により再送が発生してしまう。その場合、パケットの再送という事態に遭遇したユーザーは長時間、動画が停止する事になり、スムーズな動画の再生が困難になる。そのため、HTTP Live Streaming では、ハンドオーバーが発生する状況下ではスムーズな動画の再生が困難である。しかし、MPEG-DASH でのストリーミング再生であれば、ネットワーク環境に合わせて動画の質を変化させる事で、ネットワークの負荷を適正に保つ事ができるので、上記のような事態を防ぐ事ができる。

6.2　今後の展開

今回、MPEG-DASH として実装した手法は[37]で提案された手法のみである。そのため、今後はさらに様々な手法を実装し、比較・評価する必要があると考えている。また、今回の性能評価実験は、シミュレーション実験であり、かつ、現段階では MIMO といった実際の LTE ネットワークで用いられている技術を反映しきれていない。そのため、実際の LTE ネットワークにおいて HTTP Live Streaming や MPEG-DASH を用いて動画再生を行ってみる実機実験に今後取り組む必要がある。

参考文献

[1]YouTube[online]:http://www.youtube.com/

[2]ニコニコ動画[online]:http://www.nicovideo.jp/

[3]Andrew Fecheyr-Lippens, “A Review of HTTP Live Streaming,” Jan. 2010.

[4]HTTP Dynamic Streaming [online]:
http://www.adobe.com/jp/products/httpdynamicstreaming/

[5]野﨑 寛也, 甲藤 二郎, “HTTP Live Streaming における TCP 性能評価実験,” 電子情報通信学会技術研究報告, Vol. 110, No. 448, pp. 633-638, 2011 年 3 月.

[6]S. Shunmuga Krishnan, Ramesh K. Sitaraman,”Video Stream Quality Impacts Viewer Behavior:Inferring Causality Using Quasi-Experimental Designs,”Proceedings of the 2012 ACM conference on Internet measurement conference, Nov. 2012.

[7]Thomas Stockhammer,”Dynamic Adaptive Streaming over HTTP-Standards and Design Principles,”Proceedings of the second annual ACM conference on Multimedia systems, Feb. 2011.

[8]V. Jacobson, “Congestion Avoidance and Control,” in Proc. SIGCOMM, Stanford, CA, Aug. 1988.

[9]W. Richard Stevens, “TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms,” RFC2001, Jan. 1997.

[10]Janey C. Hoe, “Improving the Start-up Behavior of a Congestion Control Scheme for TCP,” proceedings of ACM SIGCOMM ’96 Aug. 1996.

[11]M. Mathis, J. Mahdavi, Sally Floyd, and A. Romanow, “TCP Selective Acknowledgment Options,” RFC2018, Oct. 1996

[12]H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson, “RTP: A Transport Protocol for Real-Time Applications,” RFC1889, Jan. 1996.

[13]H. Schulzrinne, A. Rao, R. Lanphier, “Real Time Streaming Protocol (RTSP),” RFC2326 Apr. 1998.

[14]甲藤二郎, “ネットワーク制御と管理(Network-Application2)”,2009

[15]HTTP Dynamic Streaming Datasheet [online]:
http://www.adobe.com/products/httpdynamicstreaming/pdfs/httpdynamicstreaming_datasheet.pdf

[16] HTTP Dynamic Streaming FAQ [online]:
http://www.adobe.com/jp/products/httpdynamicstreaming/faq/

[17]Christopher Muller,Christian Timmerer,”A Test-Bed for the Dynamic Adaptive Streaming over HTTP featuring Session Mobility,”Proceedings of the second annual ACM conference on Multimedia systems, Feb. 2011.

[18]Ricky K. P. Mok, Xiapu Luo, Edmond W. W. Chan, and Rocky K. C. Chang,"QDASH:A QoE-aware DASH system",Proceedings of the 3rd Multimedia Systems Conference, Feb. 2012
[19]Yago Sanchez,Thomas Schierl,Danny De Vleeschauwer,"iDASH:Improved Dynamic Adaptive Streaming over HTTP using Scalable Video Coding," Proceedings of the second annual ACM conference on Multimedia systems, Feb. 2011.
[20]Luca De Cicco,Saverio Mascolo,Vittorio Palmisano,"Feedback Control for Adaptive Live Video Streaming," Proceedings of the second annual ACM conference on Multimedia systems, Feb. 2011.
[21]Jenq-Shiou Leu, Cheng-Wei Tsai, and Chih-Wei Yi,"Improving Adaptive Streaming Service across Wired/Wireless Networks",Proceedings of the 2009 Tenth International Conference on Mobile Data Management: Systems, Services and Middleware
[22]Remi HOUDAILLE, Stephane GOUACHE,"Shaping HTTP adaptive streams for a better user experience,"Proceedings of the 3rd Multimedia Systems Conference, Feb. 2012
[23]Saamer Akhshabi,Ali C.Begen,Constantine Dovrolis,"An Experimental evaluation of Rate-Adaptation Algorithms in Adaptive Streaming over HTTP,"Proceedings of the second annual ACM conference on Multimedia systems, Feb. 2011.
[24]Christopher Muller, Stefan Lederer and Christian Timmerer,"An Evaluation of Dynamic Adaptive Streaming over HTTP in Vehicular Environments",Proceedings of the 4th Workshop on Mobile Video, Feb. 2012
[25]@IT サービス開始まで 1 年に迫った LTE って何？
[online]: http://www.atmarkit.co.jp/fnetwork/rensai/lte01/01.html
[26]大久保 尚人,ウメシュ アニール,岩村 幹生,新 博行,"高速・大容量・低遅延を実現する LTE の無線方式概要",NTT DOCOMO テクニカル・ジャーナル Vol.19 No.1
[27]@IT LTE を支える 3 つの要素技術
[online]:http://www.atmarkit.co.jp/fnetwork/rensai/lte04/01.html
[28]@IT どこまで出る？LTE の通信速度
[online]:http://www.atmarkit.co.jp/fnetwork/rensai/lte03/02.html
[29]西田 克利,田中 威津馬,輿水 敬,"All-IP ネットワークを実現する SAE 基本制御技術",DOCOMO テクニカル・ジャーナル Vol.17 No.3
[30] 全国 200 地点スマートフォン LTE 通信速度実測調査
[online]:http://www.ictr.co.jp/topics_20121106.html
[31]@IT LTE のネットワーク動作
[online]:http://www.atmarkit.co.jp/fnetwork/rensai/lte06/01.html
[32]Scenargie[online]:http://www.spacetime-eng.com/

[33]IP ドメインサーチ[online]:http://www.mse.co.jp/ip_domain/
[34]鈴木 啓介,森田 崇,楢葉 槙,繆 震,"LTE を収容するコアネットワーク(EPC)の開発",NTT DOCOMO テクニカル・ジャーナル Vol.19 No.1
[35]Ping!Up[online]: http://www.fatoen.de/en/
[36] tPacketCapture[online]:http://www.taosoftware.co.jp/en/android/packetcapture/
[37]Chenghao Liu, Imed Bouazizi, Moncef Gabbouj, "Rate Adaptation for Adaptive HTTP Streaming," Proceedings of the second annual ACM conference on Multimedia systems, Feb. 2011

謝辞

本研究を行うにあたり、日頃から研究に関して様々なご指導をいただきました甲藤二郎教授に心からの感謝の意を表します。
また、研究等で、アドバイスをしてくださったネットワーク班の先輩方に深く御礼申し上げます。
最後に、研究だけでなく、様々な面でお世話になりました甲藤研究室の皆様と私をここまで育ててくださった家族に、深く感謝致します。

2013年2月8日

野崎 寛也

発表文献リスト

[1]野﨑 寛也, 甲藤 二郎, “HTTP Live Streaming における TCP 性能評価実験,” 電子情報通信学会技術研究報告, Vol. 110, No. 448, pp. 633-638, 2011 年 3 月.

[2]野﨑 寛也, 甲藤 二郎, “HTTP Live Streaming における TCP 性能評価実験,” 電子情報通信学会 総合大会, B-6-31, 2011 年 3 月.

[3]野﨑 寛也, 甲藤 二郎, “Performance Evaluation of HTTP Streaming under Handover Events in LTE Networks,” 電子情報通信学会技術研究報告, 2013 年 3 月.(投稿中)

[4]Tomoya NOZAKI, Jiro KATTO, “Performance Evaluation of HTTP Streaming under Handover Events in LTE Networks,” IEEE ICC 2013 Workshops on Video Quality Driven Multimedia Streaming in Mobile Wireless Networks (投稿中)